BBN SYSTEMS AND TECHNOLOGIES
A Division of Bolt Beranek and Newman Inc

ARPA Order Number 8455
Contract Number: N00014-92-C-0035
Contract Duration: 16 April 1992 - 30 June 1995
Principal Investigators: J. Makhoul, (617)873-332
M. Bates, (617)873-3496

Final Report

USABLE, REAL-TIME, INTERACTIVE SPOKEN LANGUAGE SYSTEMS

Madeleine Bates, Robert Bobrow, Francis Kubala, Robert Ingria, John Makhoul, Scott Miller, Long Nguyen, Sandra Peters, Richard Schwartz, David Stallard, George Zavaliagkos

September 1994

Contents

List of Tables

List of Figures

Chapter 1

Executive Summary

This is the final technical report for the project Usable, Real-Time, Interactive Spoken Language Systems, sponsored by the Advanced Research Projects Agency (ARPA) and monitored by ONR under contract No. N00014-92-C-0035 (BBN Reference Number 11617) during the period 16 April 1992 to 30 June 1994.

The objective of this project was to make the next significant advance in human-machine interaction by developing a spoken language system (SLS) that operates in real-time while maintaining high accuracy on cost-effective COTS (commercial, off-the-shelf) hardware. The system has a highly interactive user interface, is largely user independent and to be easily portable to new applications. The BBN HARC spoken Language system consists of the Byblos speech recognition system and the Delphi or HUM language understanding system.

Our research has concentrated on the development of an effective SLS system (that is, the integration between speech and language processing), advances in language processing to move away from a pipeline, syntax-first architecture to one in which syntax and semantics play complementary roles, processing ill-formed input in a principled, domain-independent way, and developing a completely novel approach to language understanding.

The BYBLOS speech recognition system uses a novel four-pass search strategy. It produces ordered lists of the N top-scoring hypotheses (N-best) which are then reordered by several detailed knowledge sources. The N-best strategy [48, 3, 26, 63, 64, 65] permits the use of computationally prohibitive models by greatly reducing the search space to a few dozen word sequences. It has enabled us to use cross-word-boundary triphone models and trigram language models with ease. The N-best list is also a robust interface between speech and natural language that provides a way to recover from speech errors in the top choice

word sequence.

The Delphi language understanding system uses a definite clause grammar formalism, augmented by the use of constraint nodes [69] and a labelled argument formalism. The parsing algorithm uses a statistically trained agenda to produce the single best parse for an input utterance [20]. We have added a robust fragment parser to deal with speech errors when the correct answer is not in the N-best list [23]. We developed a new hybrid method of representation that combines the best features of logical and frame representations. One of the most important features of Delphi is the Semantic Linker, which is able to handle ill-formed input by linking partially-understood fragments on the basis of meaning, without requiring a full syntactic analysis.

A novel, potentially high-payoff, approach to language understanding was initiated under this contract. The approach, called HUM (Hidden Understanding Model) incorporates a statistical model of meaning. If successful, HUM will lead to automatic acquisition of linguistic knowledge, coupled with high performance. An initial version of HUM was implemented and tested in the Airline Travel Information Service (ATIS) domain. The test showed the basic soundness of this novel approach [46, 43].

In the area of portability, we examined several new tasks as possible targets for porting the spoken language technology, including shared-map planning, multi-media conferencing, and other database applications. We chose SPLINT, a database of information about Air Force bases and equipment, and ported the HARC SLS to that database in conjunction with another contract [6, 14].

During the period of this contract, although not as part of it, BBN released the first version of the HARK™speech recognition system as a commercial product. HARK is based in part on BYBLOS speech recognition technology. HARK is the first product of its kind to run in real-time with a large vocabulary (over 2000 words) on off-the-shelf, audio-capable Unix workstations. A companion product, the HARK Prototyper™, allows the HARK vocabulary and grammar to be configured by application programmers.

Major accomplishments achieved during this project include:

1. Designed and implemented an initial version of HUM (hidden understanding model), a new method for language understanding, based on learning a statistical model of semantics from annotated data. The system minimizes the labor-intensive writing of semantic rules and automates system training from data, resulting in a greater level of domain independence.

2. Began the development of tools and processes for porting a spoken language system to a new domain. The tools were tested by performing an actual port, under a

Rome Lab contract, to a database containing information about Air Force bases and equipment.

3. Proposed concrete steps toward defining the SemEval evaluation methodology – a domain-independent methodology for the evaluation of semantics capabilities – and developed annotation tools to facilitate explorations of SemEval.

4. Extended the ATIS speech understanding system to the ATIS3 database, increasing the vocabulary to about 3,000 words, and participated in the annual ARPA evaluations. There was a significant decrease in error rate over the previous year and the BYBLOS system again had the highest speech recognition accuracy in the evaluation.

5. Our research into the appropriate integration of lexical, syntactic, and semantic knowledge produced a system that is capable of using efficiently whatever types of knowledge will produce a valid interpretation of an utterance in context. Syntactic knowledge is used if it is available and reliable, but an utterance that is outside the scope of DELPLHI's grammar can still be understood if phrases can be recognized and combined semantically.

6. We developed a learnable model of semantics, to facilitate the acquisition of domain-specific information that was formerly very labor intensive to produce.

7. At the 1992 ARPA Speech and Natural Language Workshop, BBN gave a demonstration of the first 1000-word, real-time, continuous, speaker independent speech recognition system implemented on an off-the-shelf workstation, without any accelerator boards.

8. BBN also gave the first real-time demonstration of a complete spoken language understanding system in the ATIS (Air Travel Information System) domain.

10. We participated fully in the ATIS data collection effort, contributing a large portion of the ATIS training and evaluation data.

11. A demo suite, which includes real-time speech recognition and understanding demonstrations, was delivered to NSA and NIST. The demos run on the Silicon Graphics Indigo. The real-time ATIS system has been used by NIST to collect ATIS data from subjects.

Drs. Madeleine Bates and John Makhoul were invited to participate in the National Academy of Sciences Colloquium on Human-Machine Communication by Voice, Irvine, California, February 8-9, 1993. They gave the following presentations: Madeleine Bates," Models of Natural Language Understanding.", John Makhoul, "State of the Art in Continuous Speech Recognition.". These two papers appeared in a book published by the National Academy of Sciences[7, 45].

We have chaired and worked closely with several DARPA-wide SLS program committees, in particular the committee that determines the methodology for the ATIS systems (the MADCOW committee). Some of our contributions to the evaluation methodology are documented in [24, 73].

Dr. Madeleine Bates was Chair of the 1993 ARPA Human Language Technology Workshop, which took place on March 21-24, 1993, at the Merrill Lynch Conference Center, Plainsboro, NJ. [11] BBN presented demonstrations of the BBN real time ATIS system at this workshop, and the BBN real time ATIS system was available in the demo room throughout the workshop. She also co-chaired the Applied Natural Language Processing Conference in Trento, Italy in April, 1992.

We participated in the annual ARPA Speech and NL workshops, and the Human Language Technology workshops, by presenting papers and giving demonstrations of the technology developed under this effort. References to the papers from these workshops, as well as other presentations and papers produced under this contract, can be found in the bibliography.

In Chapter 2 of this document, we present a general overview of the BBN HARC spoken language understanding system. Chapter 3 describes the Delphi natural language component of HARC, and Chapter 4 details the Semantic Linker component of Delphi. Some of the speech research carried out under this contract is discussed in Chapter 5, and our most recent research in a novel method of natural language processing is presented in Chapter 6.

Chapter 2

Overview of HARC

2.1 Introduction

In this chapter we describe the design and performance of a complete spoken language understanding system currently under development at BBN. The system, dubbed HARC (Hear And Respond to Continuous speech), successfully integrates state-of-the-art speech recognition and natural language understanding subsystems. The system has been tested extensively on a restricted airline travel information (ATIS) domain with a vocabulary of over 1000 words. In this application, the system functions as an electronic airline guide, searching a database to answer questions posed by the user.

HARC is implemented in portable, high-level software that runs in real time on today's workstations to support interactive online human-machine dialogs at a very comfortable pace. No special purpose hardware is required other than an A/D converter to digitize the speech.

The system works well for any native speaker of American English and does not require any enrollment data from the users. HARC has shown consistently high performance in formal evaluations on the ATIS domain.

The BBN HARC spoken language system weds two technologies, speech recognition and natural language understanding, into a deployable human-machine interface. The problem of understanding goal-directed spontaneous speech is harder than recognizing and understanding read text, due to greater variety in the speech and language produced. We have made minor modifications to our speech recognition and understanding methods to deal with these variabilities. The speech recognition uses a novel multipass search strategy

that allows great flexibility and efficiency in the application of powerful knowledge sources. The natural language system is based on formal linguistic principles with extensions to deal with speech errors and to make it robust to natural variations in language. The result is a very usable system for domains of moderate complexity.

While the techniques used here are general, the most complete test of the whole system thus far was made using the ATIS corpus, which is briefly described in Section 2. Section 3 describes the techniques used and the results obtained for speech recognition, and Section 4 is devoted to natural language. The methods for combining speech recognition and language understanding, along with results for the combined system are given in Section 5. Finally, in Section 6, we describe a real-time implementation of the system that runs entirely in software on a single workstation.

More details on the specific techniques used, the makeup of the corpus, and the results can be found in the papers presented at the ARPA Workshops on Speech and Natural Language and other meetings [41, 23, 21, 44, 33, 40, 61, 46, 70, 40, 50, 62, 49, 11, 48].

2.2 The ATIS Domain and Corpus

2.2.1 ATIS Domain

The Air Travel Information Service (ATIS) is a system for getting information about flights. The information contained in the database is similar to that found in the Official Airline Guide (OAG) but is for a small number of cities. The ATIS corpus consists of spoken queries by a large number of users who were trying to solve travel related problems. The ATIS0 and ATIS1 corpora contain about 4,000 utterances, most of which were read sentences, mostly by speakers with dialects from the southern U.S. The ATIS2 training corpus consists of 12,214 spontaneous utterances from 349 subjects (159 female, 190 male) who were using simulated or real speech understanding systems in order to obtain realistic speech and language. The data originated from 5 collection sites using a variety of strategies for eliciting and capturing spontaneous queries from the subjects [44], with a disproportionate amount of the data coming from MIT.

1,289 utterances were truncated or contained word fragments due to stuttering. Many more contained various nonspeech sounds. There were also frequent long pauses and hesitations in the data. Each sentence in the corpus was classified as class A (self contained meaning), class D (referring to some previous sentence), or class X (impossible to answer for a variety of reasons). The speech recognition system was tested on all three classes, although the results for classes A and D were given more importance. The natural language system

and combined speech understanding systems were scored only on classes A and D, although they were presented with all of the test sentences in their original order.

2.2.2 Formal Evaluation Conditions

The November '92 evaluation test set has data from 35 speakers. The number of utterances per speaker varied from 2 to 41, but the number of utterances from each of the 5 data-collection sites was carefully balanced. All results given were collected with the Sennheiser microphone (same as the training data). The recognition mode was speaker-independent -- the test speakers were not in the training set and every sentence was treated independently.

By committee decision there was no common baseline control condition for the training data or speech grammar to be used for the ATIS tests. The only constraint was that the single common test set must be used.

2.3 BYBLOS – Speech Recognition

BYBLOS is a state-of-the-art, phonetically-based, continuous speech recognition system that has been under development at BBN for over seven years. This system introduced an effective strategy for using context-dependent phonetic hidden Markov models (HMM) and demonstrated their feasibility for large vocabulary, continuous speech applications [25]. Over the years, the core algorithms have been refined primarily on artificial applications using read speech for training and testing. These same basic algorithms, with small extensions, have proven to be remarkably suited to the recognition of completely spontaneous speech produced in a goal-directed task, such as ATIS.

2.3.1 New Extensions for Spontaneous Speech

Spontaneous queries spoken in a problem-solving dialog exhibit a wide variety of disfluencies. There were three very frequent effects that we attempted to solve – excessively long segments of waveform with no speech, poorly transcribed training utterances, and a variety of nonspeech sounds produced by the user.

When background noise is present, the HMM is not a particularly reliable discriminator of speech vs. silence, and many insertion errors result. We chose to find and truncate long

regions of nonspeech with a very reliable energy d speech detector that can deal with noise bursts near the speech. The speech detector s several simple adaptive SNR-dependent detection thresholds

Typically, there are many untranscribed short segments of background silence remaining in the waveforms after truncating the long ones. We found them to be so numerous that they measurably degraded the performance gain usually derived from using cross-word-boundary triphone HMMs. We devised a procedure to automatically mark the missing silence locations by running the recognizer on the training data constrained to the correct word sequence, but allowing optional silence between each word. Then we retrained the model using the output of the recognizer as *corrected* transcriptions.

Spontaneous data from naive speakers exhibits a large number and variety of nonspeech events, such as pause fillers (um's and uh's), throat clearings, coughs, laughter, and heavy breath noise. We attempted to model a dozen broad classes of nonspeech sounds that were both prominent and numerous. However, when we allowed the decoder to find nonspeech models between words, we found that there were more false detections than correct ones. Because our silence model had little difficulty dealing with breath noises, lip smacks, and other noises, our best results were achieved by making the nonspeech models very unlikely in the grammar.

2.3.2 Forward-Backward N-best Search Strategy

The BYBLOS speech recognition system uses a novel multi-pass search strategy designed to use progressively more detailed models on a correspondingly reduced search space. It produces an ordered list of the N top-scoring hypotheses which is then reordered by several detailed knowledge sources. This N-best strategy [26, 63] permits the use of otherwise computationally prohibitive models by greatly reducing the search space to a few (N=20-100) word sequences. It has enabled us to use cross-word-boundary triphone models and trigram language models with ease. The N-best list is also a robust interface between speech and natural language that provides a way to recover from speech errors.

We use a 4-pass approach to produce the N-best lists for natural language processing.

1. A forward pass with a bigram grammar and discrete HMM models saves the top word-ending scores and times [5].
2. A fast time-synchronous backward pass produces an initial N-best list using the Word-Dependent N-best algorithm [64].

3. Each of the N sentence hypotheses is rescored with cross-word-boundary triphones and semi-continuous density HMMs and the best list is reordered

4. The N-best list is rescored with a trigram grammar and reordered again.

Each utterance is quantized and decoded three times, once with each gender-dependent model and once with a gender-independent model. For each utterance, the N-best list with the higher top-1 hypothesis score is chosen. The top choice in the final list constitutes the speech recognition results reported below. Then the entire list is passed to the language understanding component for further reordering and interpretation.

2.3.3 Training Conditions

Although there were no restrictions on the training data to be used, we used speech data from the ATIS2 subcorpus exclusively to train the parameters of the acoustic model. We did this because we felt that the ATIS2 subset best represented the conditions of the test and because we felt that simply adding more training data to achieve incremental improvements is scientifically uninteresting.

However, we filtered the training data for quality in several ways. We removed from the training any utterances that were marked as truncated, containing a word fragment, or containing rare nonspeech events. Our forward-backward training program also automatically rejects any input that fails to align properly, thereby discarding many sentences with incorrect transcriptions. These steps removed 1,289 utterances from consideration.

After holding out the 1001 sentences of the Feb. '92 test as a development test set, we were left with a total of 10925 utterances for training the HMMs. For statistical language model training we used all available (17,313) sentence texts from ATIS0, ATIS1, and ATIS2 (excluding the development test sentences from the language model training during the development phase).

The lexicon used for recognition was initialized by including all words observed in the complete grammar training texts. Common closed-class words such as days of the week, months, numbers, plane types, etc., were completed by hand. Similarly, we included derivations (mostly plurals and possessives) of many open-class words in the domain. We also added about 400 concatenated word tokens for commonly occurring sequences such as WASHINGTON_D_C, SAN_FRANCISCO, and D_C_TEN. The final size of the lexicon was 1830 words. For the November '92 evaluation test set only 57 word tokens, covering

52 unique words, were out-of-vocabulary (OOV) for this lexicon. This is only a 0.6% OOV word occurrence rate over the whole test set.

We estimated the parameters of our statistical bigram and trigram grammars using a new backing-off procedure [55] that is somewhat simpler than that of Katz [38]. The n-grams were computed on word classes in order to share the very sparse training. A total of 1090 semantic classes were defined (most words remained singletons in their class).

2.3.4 Speech Recognition Results

Table 2.1 shows the official results for BYBLOS on this evaluation, broken down by utterance class. We also show the average perplexity of the bigram and trigram language models as measured on the evaluation test sets (ignoring out-of-vocabulary words).

Sentence Class	Bigram Perplex	Trigram Perplex	Feb92–Nov92–Nov93 % Word Errors
A+D	17	12	6.2–4.3–3.3
A+D+X	20	15	9.4–7.6–4.4
A	15	10	5.8–4.9–3.0)
D	20	14	7.0–4.8–4.0)
X	35	28	17.2–14.5–8.6)

Table 2.1: Official SPREC results on Feb92, Nov92, and Nov93 test sets.

The word error rate in each category was lower than any other speech system reporting on this data.

For a discussion of these results, the reader is referred to section 5.1.4 of [60].

2.4 Delphi – Natural Language Understanding

The natural language (NL) component of HARC is the Delphi system. Delphi uses a definite clause grammar formalism, augmented by the use of constraint nodes [69] and a labelled argument formalism [21]. Our initial parser used a standard context-free parsing algorithm, extended to handle a unification-based grammar. It was then modified to integrate semantic processing with parsing, so that only semantically coherent structures

would be placed in the syntactic chart. The speed and robustness were enhanced by switching to an agenda-based chart-parser, with scheduling depending on the measured statistical likelihood of grammatical rules [20]. This greatly reduced the search space for the best parse.

The most recent version of Delphi includes changes to the syntactic and semantic components that maintain the tight syntactic/semantic coupling characteristic of earlier versions, while allowing the system to provide semantic interpretations of input which has no valid global syntactic analysis. This included the development of a "fallback component" [23, 70], in which statistical estimates play an important role. This component allows Delphi to deal effectively with linguistically ill-formed inputs that are common in spontaneous speech, as well as with the word errors produced by the speech recognizer

2.4.1 Parsing as Transduction – Grammatical Relations

The Delphi parser is not a device for constructing syntactic trees, but an information transducer. Semantic interpretation is a process operating on a set of messages characterizing local "grammatical relations" among phrases, rather than as a recursive tree walk over a globally complete and coherent parse tree. The grammar has been reoriented around local grammatical relations such as deep-structure subject and object, as well as other adjunct-like relations. The goal of the parser is to make these local grammatical relations (which are primarily encoded in ordering and constituency of phrases) as readily available to the semantic interpreter as information explicitly encoded in the words themselves.

From the point of view of a syntactic-semantic transducer, the key point of any grammatical relation is that it licenses a small number of semantic relations between the "meanings" of the related constituents. Sometimes the grammatical relation constrains the semantic relation in ways that cannot be predicted from the semantics of the constituents alone (e.g. Given "John", "Mary", and "kissed", only the grammatical relations or prior world knowledge determine who gave and who received). Other times the grammatical relation simply licenses the only plausible semantic relation (e.g., "John", "hamburger", and "ate"). Finally, in sentences like "John ate the fries but rejected the hamburger", semantics would allow the hamburger to be eaten, but our knowledge of its destiny is mediated by its lack of any grammatical relation to "ate".

Grammatical relations are expressed in the grammar by giving each element of the right hand side of a grammar rule a grammatical relation as a label. A typical rule, in schematic form, is:

```
(NP    )  + :HEAD (NP ...) :PP-COMP (PP :PREP    )
```

which says that a noun phrase followed by a prepositional phrase provides evidence for the relation PP-COMP between the PP and HEAD of the NP.

One of the right-hand elements must be labeled the "head" of the rule, and is the initial source of information about the semantic and syntactic "binding state" which controls whether other elements of the right-hand side can "bind" to the head via their labeled relation

This view made it possible to both decrease the number of grammar rules (from 1143 to 453) and increase syntactic coverage. Most attachments can be modelled by simple binary adjunction, and since the details of the syntactic tree structure are not central to a transducer, each adjunct can be seen as being "logically attached" to the "head" of the constituent.

In effect, we factored single grammar rules (that produced ordered sequences of constituents) into smaller binary adjunction rules that can be combined together in various ways; ordering constraints on these adjuncts are provided by lexical semantic well-formedness rules. For example, rather than using subcategorization features to name sets of categories that appear together as complements of a verb, we have defined approximately 15 verb phrase rules that list the possible constituents that may appear as complements to a verb. These may embed within each other freely, so long as the results are semantically interpretable by the head. Similar recursive binary complement and adjunct rules are provided for noun phrases. This recursive scheme allows the adjunction rules of the grammar to be combined together in novel ways, governed by the lexical semantics of individual words. The grammar writer does not need to foresee all possible combinations.

2.4.2 "Binding rules" – the Semantics of Grammatical Relations

The interface between parsing and semantics is a dynamic process structured as two coroutines in a cascade. The input to the semantic interpreter is a sequence of messages, each requesting the semantic "binding" of some constituent to a head. A set of "binding rules" for each grammatical relation licenses the binding of a constituent to a head via that relation by specifying the semantic implications of binding. These rules specify features of the semantic structure of the head and bound constituent that must be true for binding to take place, and may also specify syntactic requirements. Rules may also allow certain semantic roles (such as time specification) to have multiple fillers, while other roles may

allow just one filler.

As adjuncts are added to a structure, the binding list is conditionally extended as long as semantic coherence is maintained. When a constituent is syntactically complete (i.e., no more adjuncts are to be added), Delphi evaluates rules that check for semantic completeness and produce an "interpretation" of the constituent.

2.4.3 Robustness Based on Statistics and Semantics

Unfortunately, simply having a transduction system with semantics based on grammatical relations does not deal directly with the key issue of robustness – the ability to make sense of an input even if it cannot be assigned a well-formed global syntactic analysis. The difficulty with standard syntactic techniques is that local syntactic evidence is not enough to accurately determine grammatical relations. A NP followed by a verb may be the subject of that verb ("John flew to Boston") or may be unrelated ("The man I introduced to John flew to Boston"). The standard solution is to find a global parse, which provides the necessary confirming evidence for the local relations it contains.

In Delphi we view standard global parsing as merely one way to obtain evidence for the existence of the grammatical relations in an input string. Delphi's strategy is based on two other sources of information. Delphi applies semantic constraints incrementally during the parsing process, so that only semantically coherent grammatical relations are considered. Additionally, Delphi has statistical information on the likelihood of various word senses, grammatical rules, and grammatical-semantic transductions. Thus Delphi can rule out many locally possible grammatical relations on the basis of semantic incoherence, and can rank alternative local structures on the basis of empirically measured probabilities. The net result is that even in the absence of a global parse, Delphi can quickly and reliably produce the most probable local grammatical relations and semantic content of various fragments.

Delphi first attempts to obtain a complete syntactic analysis of its input, using its agenda-based best-first parsing algorithm. If it is unable to do this, it uses the parser in a fragment-production mode, which produces the most probable structure for an initial segment of the input, then restarts the parser in a top down mode on the first element of the unparsed string whose lexical category provides a reasonable anchor for top-down prediction. This process is repeated until the entire input is spanned with fragments. Experiments have shown that the combination of statistical evaluation and semantic constraints produces chunks of the input that are very useful for interpretation by non-syntactically-driven strategies.

2.4.4 Advantages of Delphi's Approach

The separation of syntactic grammar rules from semantic binding and completion rules greatly facilitates fragment parsing. While it allows syntax and semantics to be strongly coupled in terms of processing (parsing and semantic interpretation) it allows them to be essentially decoupled in terms of notation. This makes the grammar and the semantics considerably easier to modify and maintain.

We believe, however, that in the long term the most important advantage is that separating syntactic rules from semantic binding leads us to a new kind of language model, based on grammatical relations, in which syntactic, semantic and lexical knowledge can be acquired by largely automatic means. We view the role of the grammar as codifying the way that tree structure provides evidence for grammatical relations. The separation between rule types will allow us for the first time to consider the effect of grammatical relations on meaning, independently of the way that evidence for these relations is produced by the parser.

One effect of this approach is to make it possible to use a hypothesized semantic interpretation of a set of tree fragments to generate a new syntactic rule. Thus, in normal operation, the primary evidence for a grammatical relation is the result of actually parsing part of an input. However, since grammatical relations between constituents entail semantic relations, if we can make an estimate of the likelihood of certain semantic relations based on domain knowledge, pragmatics, and task models, etc., it is in principle possible to use abductive reasoning to suggest likely grammatical relations, and thereby automatically propose new grammar rules.

2.5 Combined Spoken Language System

Figure 2.1 shows the components of the entire spoken language system.

The basic interface between BYBLOS and Delphi in HARC is the N-best list. In the most basic strategy, we allowed the NL component to search arbitrarily far down the N-best list until it either found a hypothesis that produced a database retrieval or reached the end of the N-best list. However, we have noticed in the past that, while it was beneficial for NL to look beyond the first hypothesis in an N-best list, the answers obtained by NL from speech output tended to degrade the further down in the N-best list they were obtained.

For the 1992 evaluation, we optimized both the depth of the search that NL performed on the N-best output of speech and how we used a number of fall-back strategies for NL text

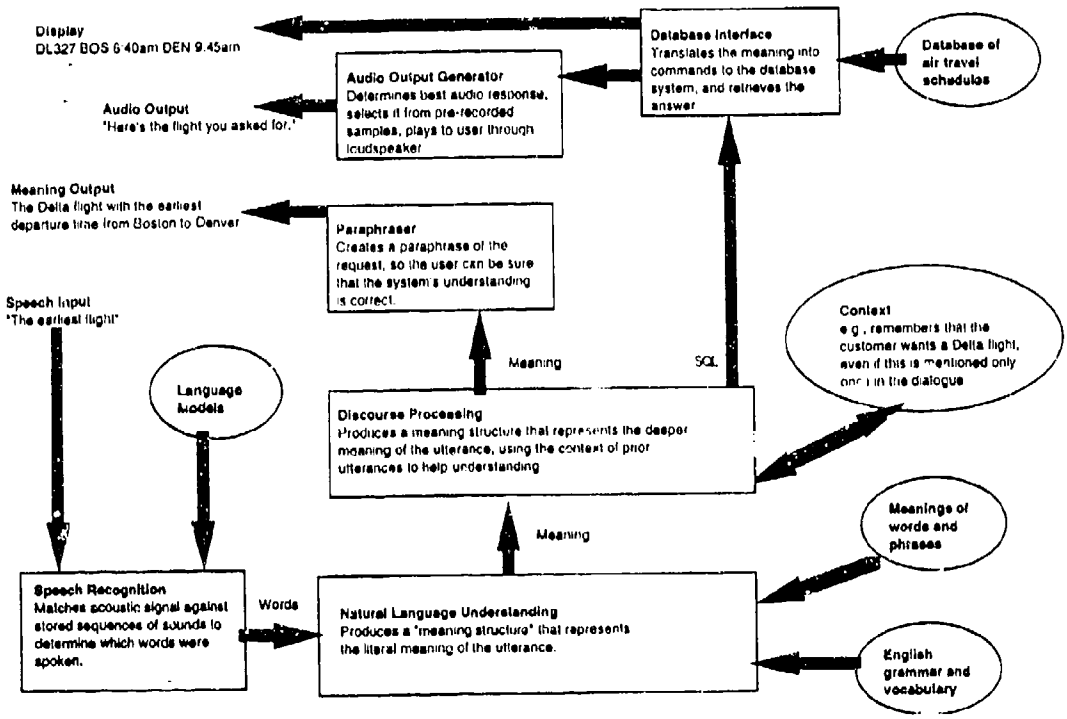


Figure 2.1: BBN's SLS ATIS System

processing [23]. We found that, given the current performance of all the components, the optimal number of hypotheses to consider was N= Furthermore, we found that rather than applying the fall-back mechanism to each of these hypotheses in turn, it was better to make one pass through the N-best hypotheses using the full parsing strategy, and then, if no sentences were accepted, make another pass using the fall-back strategy.

2.6 Interface

The interface to the BBN HARC/ATIS system consists of a few large control buttons, a sequence of status lights, and a series of display windows.

The control buttons allow the user (via the mouse) to command the system to begin to listen for a spoken question or command, to cancel a query in progress, to provide help, to clear the context, and to display another window in which the user may set various system parameters or quit.

The status lights indicate whether the system is Ready (not listening, but ready for the user to click the Listen button and talk), Listening (microphone on), Recognizing (BYBLOS speech recognition in progress) Understanding (the Delphi language understanding system in progress, followed by the translation of the understood utterance to database commands), and Retrieving (database retrieval).

The display windows include the results of the speech recognition process, Delphi's paraphrase of the meaning of the utterance, the data (usually a table) retrieved from the database, and the context that is maintained from one query to the next.

Figures 2.2 through 2.8 show the BBN ATIS screen after various kinds of queries or other user actions.

2.7 Results

In Table 2.2 we show the official performance on the November '92 evaluation data as calculated by NIST. The percent correct and the weighted error rate is given for the Delphi system operating on the transcribed text (NL) and for the combined HARC system (SLS). The results are shown for classes A+D combined and separately. The weighted error measure weights incorrect answers twice as much as no answer.

weighted error (WE) = 2 x %incorrect answers + %no answers

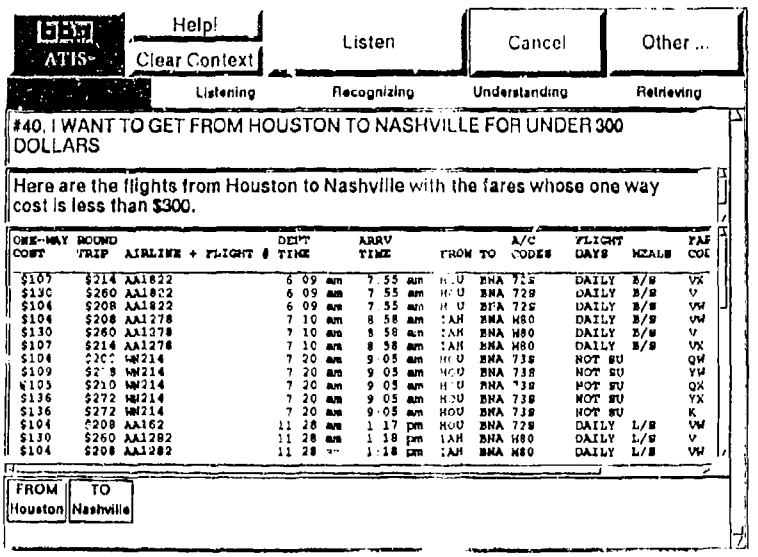


Figure 2.2: A Spoken Interaction with the BBN HARC/ATIS System

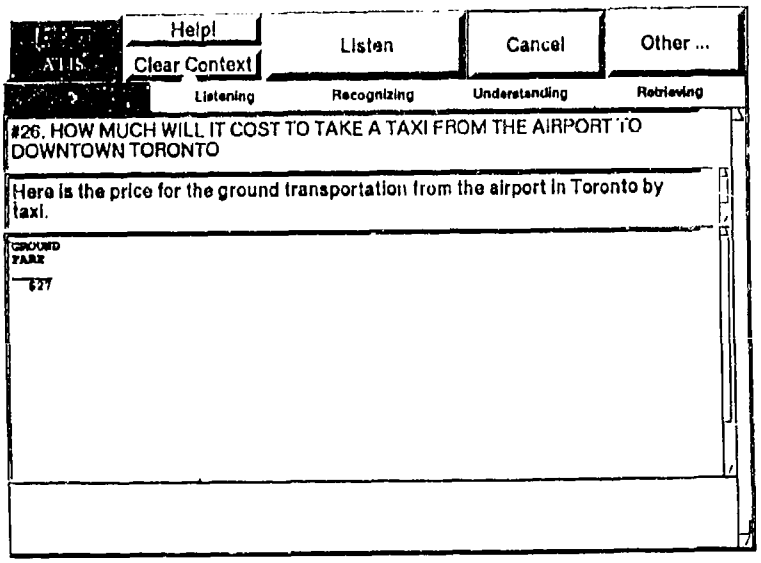


Figure 2.3: BBN/ATIS Answers a Question about Ground Transportation

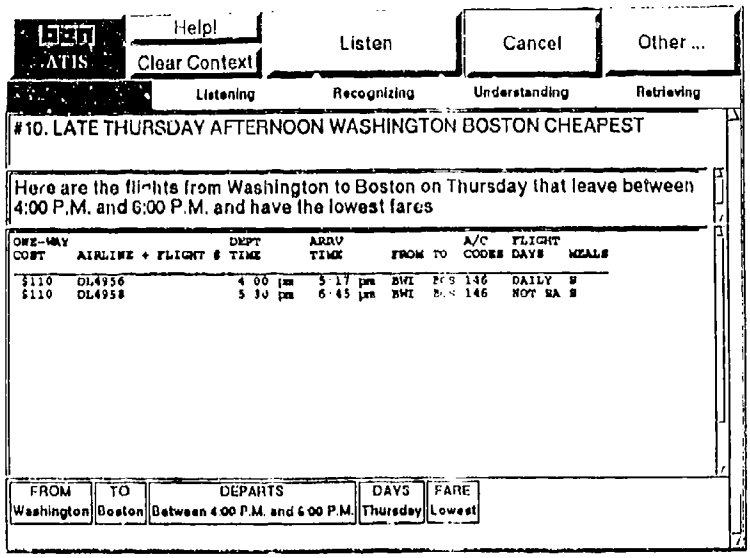


Figure 2.4: Utterances Need Not be Complete or Grammatical

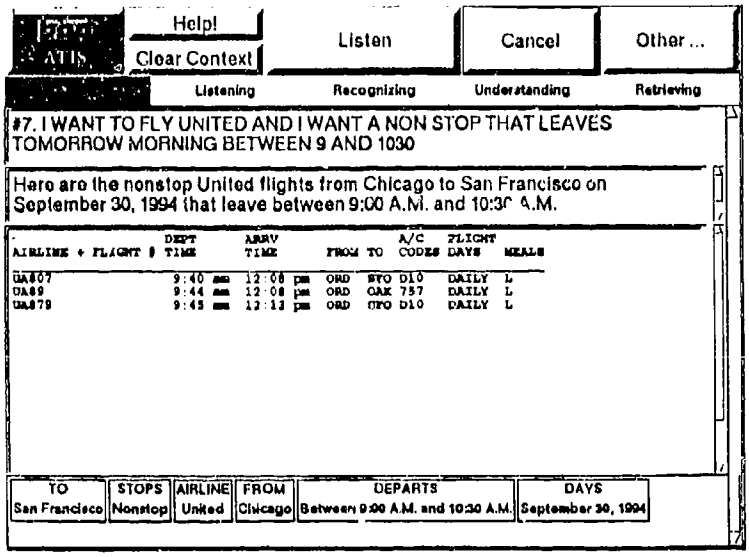


Figure 2.5: Using Prior Context, BBN/ATIS Answers a Complex Query

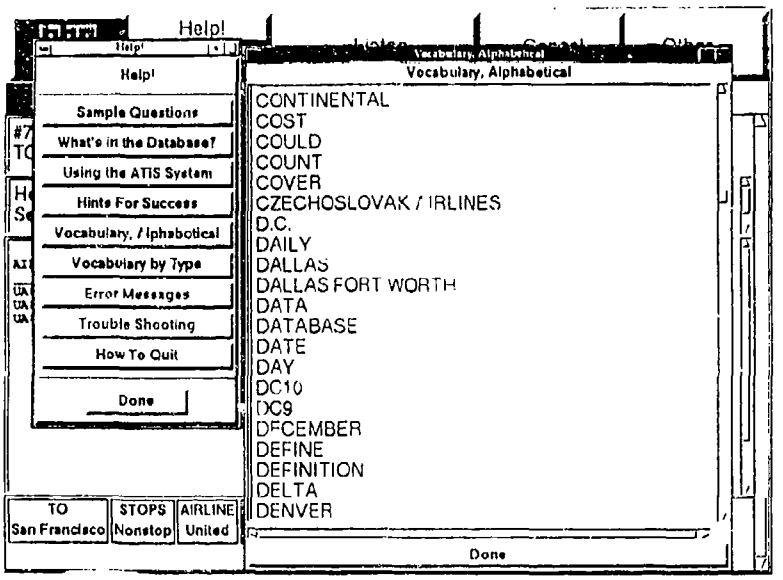


Figure 2.6: BBN/ATIS Provides a Full Vocabulary List

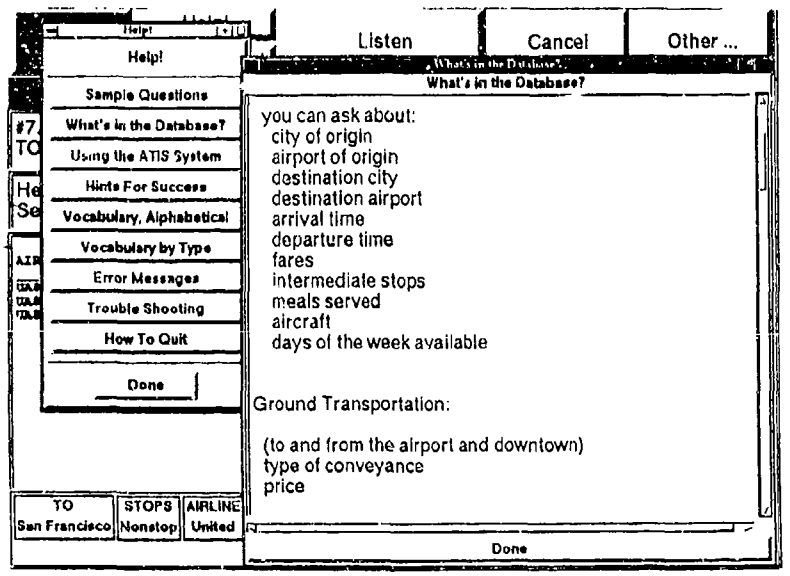


Figure 2.7: BBN/ATIS Help for the User

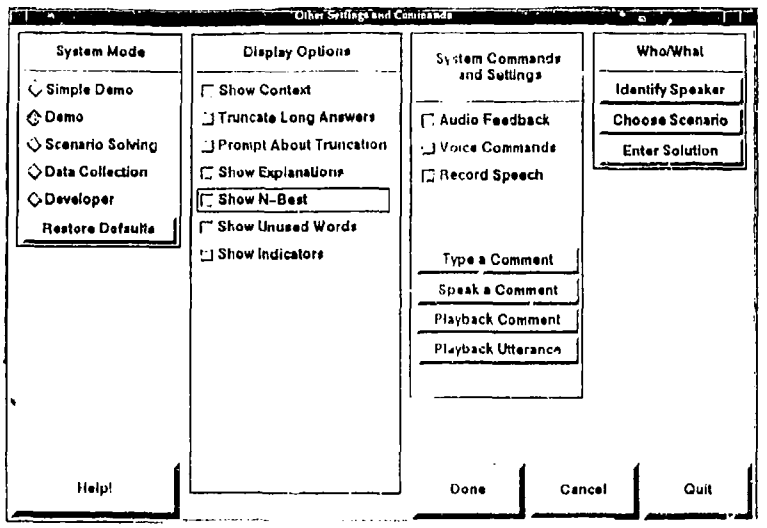


Figure 2.8: The "Other..." Window Lets User Change System Parameters

Corpus	NL Cor	NL WE	SLS Cor	SLS WE
A+D	85.0	22.0	81.0	30.6
A	88.8	15.7	84.6	23.7
D	78.6	32.8	74.9	42.5

Table 2.2: %Correct and Weighted error on the November '92 test set.

The weighted error on context-dependent sentences (D) is about twice that on sentences that stand alone (A). This higher error rate results from two phenomena. First, it is often difficult to resolve references correctly and to know how much of the previous constraints are to be kept. Second, in order to understand a context-dependent sentence correctly, we must correctly understand at least two sentences, which is less likely.

The weighted error from speech input is from 8%-10% higher than from text. However, the difference is lower than might be expected. Even though the BYBLOS system misrecognized at least one word in 25% of the utterances, the Delphi system was able to recover from most of these errors through the use of the N-best list and fallback processing.

The official SLS result for HARC was a weighted error of 30.6%. This represents a substantial improvement in performance over the weighted error during the previous (February '92) evaluation, which was 43.7%. It was the third best overall result for a spoken language system of the seven sites participating. Based on end-to-end tests with

real users, the system is usable, given that subjects were able to accomplish their assigned tasks. However, we believe that the largest remaining improvement will not be from speech modeling or basic natural language understanding, but from more careful task modeling.

After the 1992 evaluation, the official ARPA scoring metric was changed from weighted error to simple unweighted error (percent incorrect + percent unanswered).

The results for the December 1993 ARPA evaluation are given in Table 2.3.

Corpus	NL Cor	NL Err	SLS Cor	SLS Err
A+D	85.3	14.7	75.4	17.5
A	90.4	9.6	86.2	13.8
D	78.1	21.8	77.5	22.5

Table 2.3: %Correct and Simple Error on the December '93 test set.

2.8 Real-Time Implementation

A real-time demonstration of the entire spoken language system described above has been implemented. The speech recognition was performed using BBN HARK^{TM}, a commercially available product for continuous speech recognition of medium-sized vocabularies (about 1,000 words). HARK stands for High Accuracy Recognition Kit. HARK^{TM} (not to be confused with HARC) has essentially the same recognition accuracy as BYBLOS but can run in real-time entirely in software on a workstation with a built-in A/D converter (e.g., SGI Indigo, SUN Sparc, or HP715) without any additional hardware.

The speech recognition displays an initial answer as soon as the user stops speaking, and a refined (rescored) answer within 1–2 seconds. The natural language system chooses one of the N-best answers, interprets it, and computes and displays the answers, along with a paraphrase of the query so the user can verify what question the system answered. The total response cycle is typically 3–4 seconds, making the system feel extremely responsive. The error rates for knowledgeable interactive users appears to be much lower than those reported above for naive noninteractive users.

2.9 Summary

In this chapter, we have described the HARC spoken language understanding system. HARC consists of a modular integration of the BYBLOS speech recognition system with the Delphi natural language understanding system. The two components are integrated using the N-best paradigm. We have shown that this paradigm is a modular, efficient, and effective method for integrating speech recognition and language understanding components. In addition, the N-best strategy was shown to be useful within the speech recognition system as a means of applying expensive knowledge sources, such as cross-word acoustic models and trigram language models. The entire system has been implemented to run in real time on a standard workstation without the need for any additional hardware.

Chapter 3

The Delphi Natural Language Understanding System

3.1 Introduction

This chapter presents Delphi, the natural language component of the BBN Spoken Language System. Delphi is a domain-independent natural language question answering system that is solidly based on linguistic principles, yet which is also robust to ungrammatical input. It includes a domain-independent, broad-coverage grammar of English. Analysis components include an agenda-based best-first parser and a failback component for partial understanding that works by fragment combination. Delphi has been formally evaluated in the ARPA Spoken Language program's ATIS (Airline Travel Information System) domain, and has performed well. Delphi has also been ported to a spoken language demonstration system in an Air Force Resource Management domain. We discuss results of the evaluation as well as the porting process.

Delphi is a natural language understanding system based on general linguistic principles which is adaptable to any question-answering domain. It incorporates a number of domain-independent knowledge bases, including a general, broad-coverage grammar of English with a powerful and flexible handling of complementation. Unlike most other linguistically motivated systems, however, Delphi is also highly robust, allowing for partial understanding when an input is ungrammatical, disfluent, or not properly transcribed by a speech recognizer. Thus, Delphi can be used for a spoken language application as readily as for a written one. Furthermore, Delphi's partial understanding component, called the Semantic Linker, is driven off the same system of semantic rules as Delphi's regular

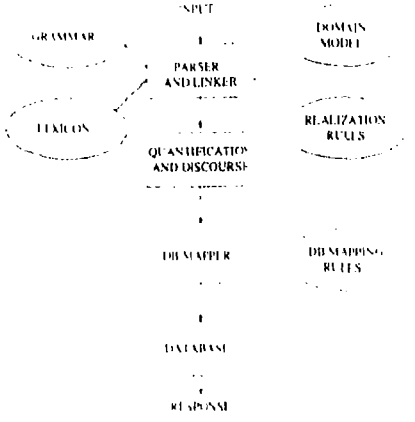


Figure 3.1: System Diagram

best-first parser. Building a robust application therefore requires no additional effort.

There are several components of the system, which is diagramed in Figure 3.1.

First are the parser and Semantic Linker, which output an intermediate representation we call a "semantic graph". The semantic graph is passed to a quantification stage which produces a fully scoped logical form from it. The logical form is then passed to the discourse stage, which resolves pronominal references and performs other types of task-dependent constraint resolution to produce the final logical form. The final logical form is then passed to the backend translator, and then to the application system which produces the response. Several knowledge bases are employed by these analysis components, including grammar, "realization rules" and the domain model, which represents the set of classes and binary relations of the given application domain.

Delphi differs from most other linguistically motivated systems in the role that is played by syntax. The primary function of Delphi's parser and syntactic knowledge bases is not to produce a parse tree, but rather to constrain the search for an appropriate semantic graph interpretation of the utterance. Semantic graphs are produced not by rule-to-rule compositionality, but by what might be called "relation-to-relation" compositionality – the association of grammatical relations in the syntactic structure with semantic relations in the semantic graph.

This more incremental view of the syntax/semantics interface has three crucial advantages. First, there is much more flexibility with respect to ordering and optionality of constituents. Second, because relation-to-relation translations are simple, the task of porting the system

is greatly simplified. Third and finally, partial or fragmentary analyses can be represented, and a complete semantic graph interpretation for the utterance produced even when a complete syntactic analyses is not available.

In the remainder of this chapter, we describe Delphi's main processing components, representational formalisms, and knowledge base.

3.2 Grammar And The Syntax/Semantics Interface

The Delphi grammar is a broad coverage, domain independent grammar of English written in a version of the Definite Clause Grammar formalism [53] that has been extended to include labeling of right-hand side elements with the grammatical relations they bear to the head of the construction. An example is:

```
(S ?arg ?mood)
->
subject: (NP ?arg ?mood etc.)
head:    (VP ?agr ?mood etc.)
```

In this rule, there is a head VP and an NP which bears the SUBJECT relation to it. Other grammatical relations include the familiar DIRECT-OBJECT and INDIRECT-OBJECT as well as the prepositions, such as TO, FROM, WITH and so on.

Annotating sub-constituents with grammatical relations regularizes the syntactic structure with respect to particular grammatical rules, and allows a "relation-to-relation" form of compositionality, as opposed to the more traditional "rule-to-rule" version that is exemplified by such systems as Gemini [28] and the Core Language Engine [2]. In relation-to-relation compositionality, each grammatical relation in the syntactic structure corresponds to a semantic relation in a parallel semantic structure we call a "semantic graph". The terminal nodes of the semantic graph are the word meanings, corresponding to the lexical heads of syntactic structure.

An example of a semantic graph, representing the meaning of "What flights fly from Boston to Denver", may be seen in Figure 3.2. The semantic graph is not a fully quantified formula; rather it may be thought of as a form of predicate-argument representation, with quantifiers in place, from which a fully quantified formula can be generated. The allowed class and relation labels come from the domain model.

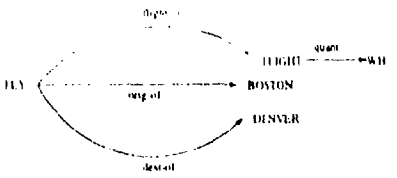


Figure 3.2: Semantic Graph

This view of the syntax/semantics interface has marked advantages. For one thing, because the syntactic/semantic structure is built up one argument at a time, it becomes much easier to accomodate such phenomena as order-variation and optionality of arguments that are difficult for other approaches

The importance of this feature may be seen in the examples of argument order-variation and optionality that abound in real data. Consider the following from the ATIS domain, in which complements can vary freely in order:

What flights fly from Boston to Denver?
What flights fly to Denver from Boston?

or be separated from the head by a modifier typically regarded as an adjunct:

What flights fly at 3 pm from Boston to Denver?

In some cases, modifiers can be omitted, as in:

What flights fly from Boston?
What flights fly to Denver?

and sometimes the omission of an argument can have anaphoric consequences, as in:

What restrictions apply?

which cannot be felicitously uttered except in a context where there is something in the discourse that a restriction could "apply" to.

Conventional approaches to subcategorization, such as Definite Clause Grammar [53], Categorial Grammar [1], PATR-II [66], and lexicalized TAG [58] all deal with complementation by including in one form or another a notion of "subcategorization frame" that specifies a sequence of complement phrases and constraints on them. Handling all the possible variations in complement distribution in such formalisms inevitably leads to an explosion in the number of such frames, and a correspondingly more difficult task in porting to a new domain.

In our approach, on the other hand, it becomes possible to view subcategorization of a lexical item as a set of constraints on the outgoing arcs of its semantic graph node. Different types of constraints – order of arguments, optionality of arguments, semantic-class constraints and semantic effects of arguments – can all be represented separately, instead of enumerating all possible argument sequences in a set of alternative subcategorization frames.

Subcategorization constraints in Delphi are encoded in lexical entries using a structure called a "map" [71]. Below is part of the lexical entry for "fly" in the ATIS domain:

```
FLY
subject: FLIGHT-OF
to:      DEST-OF
from:    ORIG-OF
completion: (and (filled flight-of)
                 (or (filled dest-of)
                     (filled orig-of))
```

Map entries have "translation", "realization" and "completion" components. The translation part of this entry specifies that the lexical head "fly" is to correspond to a semantic-graph node labeled with event-class FLY. The realization part of the entry specifies what grammatical relations the lexical item takes, and what semantic relations these correspond to, or "realize", in the semantic graph. Here, the entry specifies that "fly" takes SUBJECT, TO, and FROM complements, and that these grammatical relations correspond to the semantic relations FLIGHT-OF, DEST-OF, and ORIG-OF respectively. Semantic selectional restrictions in these argument positions – that the filler of DEST-OF be a city, for example – are implicit from the declarations of the relations in the domain model.

The "completion" part of the entry specifies what outgoing arcs are *required* for the node. Here, the entry requires that the FLIGHT-OF role be filled, and that either the DEST-OF or ORIG-OF roles be filled (forbidding the intransitive "the flight flies"). More complex optionality cases are encoded with other completion predicates. For example, the case

where an anaphor must be present ("What restrictions apply") is encoded by the predicate FILLED-OR-ANAPHOR.

Some realization rules are tied to semantic classes rather than lexical translations, and require for their application only that semantic class restrictions implicit from the domain and range of the realized relation be satisfied. Typical examples are the rules governing noun modifier meanings, such as "Delta flights", "Delta's flights", "the flights on/aboard Delta". These would all be handled by the global realization rule:

{NOM-COMP POSS ABOARD ON ...}
→
AIRLINE-OF

Determining what semantic relation a given grammatical relation instance corresponds to is most generally viewed as a form of goal-solving in Delphi, in which a chain of rules can be invoked. For example, syntactic constructions such as "X with Y", "X has Y" and "X's Y" are interpreted by first appealing to a rule mapping them to a pseudo-relation called GENERALIZED-POSSESSION, and then seeking a realization for it that is compatible with the classes of X and Y. This avoids having to write three different versions of the same realization rule.

An important advantage of the realization rule formulation, apart from its its power and flexibility, is its simplicity. Realization rules are very simple to write, and make maximal use both of knowledge about the domain and general knowledge of language.

3.3 Ill-Formedness Handling: The Semantic Linker

When an utterance cannot be parsed with Delphi's best-first parser [20] – either because it is ill-formed, mis-recognized by the speech system, or simply because it is outside the coverage of the grammar – it can still be partially understood by the system, often well enough to give the correct response. The component responsible for partial understanding in the Delphi system is called the Semantic Linker [70].

After a parse fails there is a set of fragmentary constituents left over in the chart, corresponding to a set of semantic graphs. The Semantic Linker seeks to connect these sub-graphs into a single connected one by adding links between nodes in the different sub-graphs.

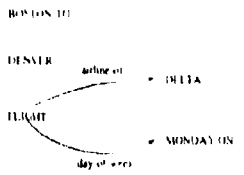


Figure 3.3: Fragment Graphs

At top-level, this is the same thing that the parser and grammar do. The difference is that the parser and grammar have an idea of what the grammatical relationship between constituents is, based on requirements of their proximity in the string and other syntactic evidence. The Semantic Linker does not have these requirements, being a looser form of combination that can ignore fragment order and skip over intervening, unanalyzable material with ease.

Although it is a very different algorithm, the Semantic Linker uses the same set of realization rules that drives the regular parser. Using the realization rules, the Linker determines for each pair of nodes in different semantic graphs the set of all links which can connect them. It then uses an A* search to find the most plausible set of links which produce a complete graph.

Suppose for example, we have the three fragments "to Boston", "Denver" and "Delta flights on Monday". Then the three corresponding sub-graphs are those shown in Figure 3.3 where a PP is treated as its NP object with the preposition as a tag. For this set of fragmentary sub-graphs, the possible links are:

```
1a.  FLIGHTS1--- DEST-OF -> BOSTON:TO
1b.  FLIGHTS1--- ORIG-OF -> BOSTON:TO

2a.  FLIGHTS1--- DEST-OF -> DENVER
2b.  FLIGHTS1--- ORIG-OF -> DENVER

3a.  DENVER--- NEARBY-TO -> BOSTON:TO
```

where the links are grouped together in a ordered list according to the fragment-pairs they connect.

The plausibility of a given link is a function of a number of different features, including penalties from assumptions made in its computation (e.g. that a given preposition can be ignored or assumed) and empirically determined probabilities for the given link (e.g. that

given an AIRLINE and a FLIGHT they are most probably linked by the relation AIRLINE-OF).

The Semantic Linker may also "hallucinate" a new node to bridge two fragments between whom no links can otherwise be computed. For example, for the utterance "from Boston to Denver", which has no explicit FLIGHT-object, a FLIGHT node can be inserted between the fragments to make sense of the utterance.

Because the Semantic Linker uses the same set of realization rules as the rest of the system, when the system is ported to a new domain the Semantic Linker can be used immediately -- a distinct advantage over some other approaches to fallback understanding, such as [23] or [36].

In formal experiments (as we discuss subsequently) the Semantic Linker has been show to dramatically improve Delphi's performance.

3.4 Quantification

The quantifier scoping module in Delphi takes a semantic graph and produces a fully-scoped expression in the logical language FMRL. The basic strategy for quantifier scoping is a descendant of that used in the LUNAR system [77]. This is made possible by the use of the semantic graph as a common underlying representation for both the grammatical and ill-formed parts of fragmentary utterances. Delphi's scoping module traps quantifiers from relative clauses, makes the quantifiers from PPs etc. outscope the NP quantifier, and resolves the scope of quantifiers from parallel constituents in terms of left-to-right order in the input. These general rules are modified to take into account differing strengths of quantifiers such as EACH.

Left-to-right ordering and syntactic structure for grammatical portions of the utterance are recovered from the semantic graph by backpointers to the lexical items and grammatical relations from which the graph was produced. Links established by the Semantic Linker are treated by the quantification mechanism as if the constituency is indeterminate, so that only left-to-right scoping rules and individual quantifier preferences take effect.

The resulting mechanism is robust, and quantificational scoping has been an insignificant source of error in the official ARPA blind-test evaluations of the ATIS system. More complex strategies have been proposed and implemented in the last two decades, and could in principle be modified to work with ill-formed input, but the simple and robust LUNAR approach handles essentially all the phenomena seen in the tens of thousands of sentences

of ATIS training collected during experiments with non-linguist users.

3.5 Discourse

The discourse mechanism of Delphi consists of several components: resolution of local ambiguities, pronominal and deictic antecedent resolution, ellipsis handling and discourse constraint propagation.

The most common case of local ambiguity in the ATIS domain involves temporal phrases as in "the nine o'clock flight". The resolution mechanism searches both for linguistic information in the current and previous sentences, as well as properties of entities in previous answers, to resolve whether "nine o'clock" is AM or PM.

The pronoun/deictic resolution mechanism used in Delphi makes use of locally expressed or implied semantic constraints to search through a set of candidate antecedents. The current mechanism ignores syntactic number as a cue, because empirically in the ATIS corpus (and we suspect in other spontaneous speech applications) it is often in error. A simple-minded focus component is used, primarily based on recency, and secondarily based on grammatical relations within an utterance. Because of the strength of semantic cues and the prevalence of ill-formed input, the use of syntactic cues for focus is limited.

The interpretation of later sentences often must include information from previous sentences, without explicit linguistic cues. This is especially true in "design dialogues", where the goal is to find a description of a set of objects that will meet some set of implicit or explicit constraints. Consider for example the following discourse from the ATIS domain.

Show Delta flights from Boston to Dallas tomorrow.
Can I leave in the morning?
Is there a nonstop flight?
Show me the American flights.
I want to go from Dallas to Chicago on Wednesday

Note that the constraints of prior sentences (such as on airline, origin, destination etc.) are implicit for subsequent sentences unless contradicted by information in the current sentence (e.g. "American" overrides the "Delta" from the first sentence) or until there is evidence that a new problem is being solved (the new origin and destination in the last sentence indicates that all previous constraints can be dropped). Delphi has a "context tracker" that

maintains a stack of the constraints from previous utterances, and has a set of rules for when constraints are to be modified or deleted before being merged with the current sentence.

Finally, we handle ellipsis as a special case of semantic linking. If we have the two utterances:

Show me the meals on the morning flight.
on American at 12:30

We can treat these as if they were one run-on ill-formed input and link "American" to "flight", and replace "morning" with "12:30", using a minor variant of the Semantic Linker linker which allows for later constraints to overwrite earlier ones of the same type. This strategy has been very effective, and covers a large class of elliptical constructions.

3.6 Backend Mapping

In order to get a response to a user query, the complete FMRL interpretation of an utterance must be translated to an expression of a target query language which can be evaluated directly against the tabular database to retrieve the answer.

A key step is bridging the gap in conceptual vocabulary between the two representations. For example, the FMRL interpretation of the query "How many flights on Delta serve meals" has one-place predicates like FLIGHT and AIRLINE, and two-place predicates like AIRLINE-OF and MEAL-OF. The database for the ATIS domain, on the other hand, only has a single table FLIGHT with fields containing airline and meal information. Delphi bridges this gap between representations with a system of local mapping rules which translate the one- and two-place predicates of the FMRL into expressions of a relational algebra target language which retrieve the extensions of these predicates.

Sometimes, however, some combination of FMRL predicates has a correspondence in the database but the individual predicates themselves do not. For example, in the database for the SPLINT domain a table relating aircraft-types to their physical characteristics has a field for the *number* of engines the aircraft has, but no representation for the engines themselves. If we now ask "How many engines does an F-16 have?", there is no local translation of the FMRL predicate ENGINE.

To deal with this, Delphi has a system of global transformations that are applied first,

rewriting subsets of the FMRL clauses to a form that can be handled with local translation. The rule that handles this example is:

```
(is-a :e engine number)
(aircraft-engine-of :a :e)
→
(is-a *count* number)
(eq (number-engines-of :a) *count*)
```

3.7 Interface To A Speech Recognizer

In spoken language applications, Delphi is interfaced to the output of the Byblos speech recognition system [9]. The N-best paradigm is used, in which the recognizer outputs in order its top N guesses at the transcription of the sentence, for some value of N (usually 5). Delphi then runs over these transcriptions in the order they have been ranked, first with the Semantic Linker disabled so that only grammatical utterances are allowed, and if none is found, runs over them again with the Semantic Linker enabled.

3.8 Results Of Formal Evaluation On ATIS

Our complete system including the Semantic Linker was evaluated in the December 1993 ARPA ATIS evaluation. Prior to evaluation, ATIS versions of the system's domain model, lexicon and realization rules had been developed using several thousand utterances of training data collected from users of ATIS. An approximately 1000-utterance set was held aside as a blind test set on which all participating sites were evaluated.

Error rate in this evaluation was defined as F+NA, where F was the percentage of queries answered incorrectly, and NA the percentage of queries not answered at all. There were two evaluations on the same corpus using this metric: one of NL text understanding alone, and the other of a complete spoken language system (SLS) comprised of Delphi and the Byblos recognizer. Our system achieved an official result of 14.7% on the NL test, which was the third-lowest error rate achieved. The SLS error rate was 17.5%.

Our own experiments show that using the Semantic Linker reduced our system's error rate on the NL test by 43%. This was largely achieved by dramatically lowering the no-answer rate NA from 18.7% to 2.3%. Just over 80% of this increment of sentences answered were answered correctly, so the Linker showed considerable accuracy.

3.9 Porting Delphi to the SPLINT Domain

Although the language understanding technology that is Delphi has greatly improved (as evidenced by objective ARPA evaluation) and has been incorporated into real-time demonstrations, there has been comparatively little effort to make systems truly transferable to various types of application systems and domains, and to develop and optimize human-machine interface paradigms for dual-use applications.

Transferability, also called portability, is key to the development of robust, practical, usable systems and, thus, to making a wide variety of applications truly practical. Portability has long been a goal ([17, 16, 34]) but seldom has been achieved.

Delphi has been developed under the premise that as much general linguistic knowledge as possible should be built into the system in a domain-independent way, modularizing the domain-dependent information to knowledge bases and easily replaceable components. By making appropriate use of general linguistic patterns, Delphi enormously reduces the amount and complexity of knowledge needed to install a new domain.

The SPLINT (Speech and Language Integration) domain was made available under a separate contract with Rome Laboratory. Development of portable modules and tools to assist the porting process were done under this contract; the actual porting was carried out under the Rome contract.

The SPLINT domain is concerned with Air Force units and their component aircraft, weaponry and other physical attributes of aircraft, ordnance, and facilities (such as air bases, runways, bunkers, etc.). It may be considered a resource management domain, with a relational database at its heart. The SPLINT database has 106 fields in 23 tables.

By studying the database, developers were able to create an initial corpus of sample questions that might be asked about the data. Some additional queries were provided by Rome Laboratory. The original set of questions was augmented by including variations that differed by substituting different words of the same type (for example, different missile names). In this way, an initial text corpus of more than 9000 sentences was created.

Some example utterances in the SPLINT domain are:

What is the range of the AGM-65C maverick missile
How many air force bases are there in the US Military Area
What are they
Where are the units with aircraft that carry vulcan tail cannons stationed
Show me a map of Griffiss
Which runways there are operational
What's the length of the longest runway
Sort the air to air missiles by range

This corpus was abstracted into query schema, so that we could more easily examine it for completeness:

```
(ARE THE <DESCRIPTION-PL> AT
        (AFB-DESCRIPTION) OPERATIONAL)

(AFB-DESCRIPTION) =
        <AFB-NAME>
        <AFB-NAME> AIR\^FORCE\^BASE
        <AFB-NAME> AFB
```

```
<AFB-NAME>  =
        GRIFFISS
        LANGLEY
```

The purpose of the query schema was NOT to provide a complete representation of all the questions that could be asked of the system, but rather to make it easier to be sure that the training set contained a wide range of representative queries that contained an appropriate distribution of entities.

In order to port Delphi to the SPLINT domain, SPLINT-specific versions of the domain model, lexicon, realization rules and db-mapping rules were needed. For the speech-understanding part of the application, word pronunciations were also necessary, as well as word-class membership for a statistical n-gram class grammar. Delphi includes "core" versions of some of these knowledge bases: a core domain model with common classes like NUMBER and TIME-OF-DAY and relations like GREATER a core lexicon with closed-class items such as prepositions as well as words appropriate to question-answering in general such as "show", to which domain-specific items have to be added.

In porting to SPLINT, 60 classes and 65 relations were added to the domain model. 400 words were added to the lexicon. Of these, approximately half were derived from database field values. 118 realization rules were added.

The domain model was built by a combination of bottom-up (database-structure driven) and top-down (corpus driven) techniques. The initial corpus was annotated for surface meaning using a variant of the notation being developed by the ARPA community for semantic evaluation. This make it possible to determine the set of concepts and relations corresponding to the linguistic expressions represented in the corpus.

The grammar did not need to be modified, with the exception of adding one rule (for constructions such as "Mach 1").

The entire process took about a person month to get 90% coverage on a 1400 sentence corpus, developed independently by a non-NL person. An additional person week was required to develop the speech-related knowledge bases. A complete spoken language system with Delphi as the understanding component, plus a Motif-based user interface, was successfully demonstrated at the 1994 ARPA Human Language Technology meeting, and at Rome Labs in New York. The porting process is described in more detail in [14, 6].

This effort demonstrates that, given an appropriate system design, it is possible to build a complete spoken language system that is robust to speech and production errors, and to do

so rapidly and straightforwardly.

3.10 Conclusion And Summary

In conclusion, we have developed a technology that makes maximal use of general linguistic knowledge to improve portability, while at the same time maintaining robustness in the face of the type of input one can expect from a real-life spoken language application. The system has been shown to reach high levels of performance in objective blind-test evaluation on the ATIS domain. The system has also been shown to be rapidly portable to a new domain, SPLINT. This did not require any changes in the underlying system code, and was done with a relatively small effort.

This work shows that computational linguistic methods, based on general knowledge of language, can be used in large, robust spoken language systems, and that special-purpose NL understanding systems do not have to be built for each new task.

Chapter 4

The Semantic Linker

4.1 Introduction

This chapter presents the Semantic Linker, a new mechanism for understanding ill-formed input. The Linker is the domain-independent fallback understanding component used by the natural language component of our spoken language system. The Semantic Linker is invoked when our regular parser is unable to parse an input; it produces a semantic interpretation by combining the fragmentary sub-parses left over in the chart using an A*-style search algorithm driven by empirically determined probabilities and parameter weights. The Semantic Linker also provides a novel method of handling ellipsis. The Semantic Linker was used in the ARPA December 1993 ATIS evaluation, where it reduced our system's error rate on the NL test by 43% - from 31.1% to 17.8%. This was one of the lowest error rates of any system tested.

An important problem for natural language interfaces is coping with input which cannot be handled by the system's grammar. A system which depends on its input being grammatical (or on lying within the coverage of its grammar) simply will not be robust and useful. Some sort of "fallback" component is therefore necessary as a complement to regular parsing.

This chapter presents the Semantic Linker, the fallback component used by the natural language component of our spoken language system. The Semantic Linker is invoked when our regular chart-based unification grammar parser is unable to parse an input; it attempts to come up with a semantic interpretation by combining the fragmentary sub-parses left over in the chart. The Semantic Linker was used in the ARPA December 1993 ATIS evaluation,

where it reduced our system's error rate on the NL test by 43% (from 31.1% to 17.8%).

Our work is motivated by the observation that syntax, by itself, is only useful insofar as it helps us in coming up with a semantic interpretation. Unlike such proposals as [42], [23] or [72], we do not attempt to "fix" the parsing process or reconstruct a parse tree from fragments, but instead try to directly produce an intermediate predicate-argument structure we call a "semantic graph". A set of fragments corresponds to a disconnected set of semantic graphs, and the task of interpretation is to find a set of links - binary relations - that connect these disconnected semantic graphs into a single connected one. Empirically determined probabilities and parameter weights are used in a A*-style best-first search to find the most plausible set of connections.

The Semantic Linker also differs crucially from proposals such as [36] in that it does not rely on task-model templates to solve the ill-formedness problem. It is therefore completely domain-independent and can be used for any task. In addition, the Semantic Linker provides a novel method of ellipsis resolution which is integrated with fallback understanding.

We devote the next section, Section 2, to a more detailed description of semantic interpretation and fragment-generation. Section 3 will discuss how the space of all possible links and associated probabilities are generated. Section 4 shows how we efficiently search this space to produce a connected semantic graph interpretation. Section 5 discusses subsequent processing, and Section 6 presents formal evaluation results and our conclusions and future plans.

4.2 Generating and Interpreting Fragments

The semantic interpretation of any constituent in our system, whether a fragment or a complete sentence, is represented by a "semantic graph", in which the nodes are the semantic interpretations of lexical heads and the links are the semantic relations between them. For example, "Delta flies a 747 to Denver" is represented by the semantic graph:

```
   /-------AIRCRAFT-OF -> 747
 FLY---- AIRLINE-OF -> DELTA
   \-----DEST-OF -> DENVER:TO
```

A PP, such "to Denver" in this example, is represented by the semantic graph representation of its NP object, tagged by the preposition of the PP. Quantifiers are left in place to be pulled out and scoped by later processing.

The semantic graph for an utterance is incrementally computed from the syntactic analysis of the utterance using a system of "realization rules" which map the grammatical relation an argument bears to the head onto the semantic relation the interpretation of the argument bears to the interpretation of the head. There are grammatical relations corresponding to the familiar notions of subject and direct-object etc., as well as to preposition-tags like "to" above. In our example, the "TO" grammatical relation holds between "flies" and "Denver". It is mapped to the DEST-OF semantic relation by the realization rule:

```
TO -> DEST-OF
```

where the semantic type requirements on head and PP object are implicit from the definition of the relation. A different set of realization rules is used for each domain.

When a complete parse of an utterance cannot be performed, we are left with a set of fragmentary analyses in the chart which correspond to syntactically well-formed and semantically coherent portions of the input string. The fragment-generation stage of the Linker extracts the most plausible fragment sub-parses associated with the longest sub-strings of the input, using probabilities associated with the grammar rules [20]. It uses a "greedy" algorithm which first chooses from the chart the coherent fragment spanning the longest sub-string of input, then the coherent fragment spanning the longest sub-string disjoint from the first sub-string, and so on, until a set of fragments has been generated that spans the entire input string.

Each fragment has an associated semantic graph as its interpretation. Suppose for example, we have the three fragments "to Boston", "Denver" and "Delta flights on Monday". Then the three corresponding sub-graphs are:

```
POSTON:TO

DENVER

FLIGHTS1------AIRLINE-OF ->  DELTA
   \--------- DAY-OF-WK -> MONDAY:ON
```

This set of n sub-graphs can be turned into a complete connected graph, and therefore a complete interpretation of the utterance, if we can find a set of $n - 1$ new links between nodes in different sub-graphs.

4.3 Computing the Possible Links and Their Probabilities

As a first step in finding the best set of connecting links, the Semantic Linker computes the "link-group list", which has one element, or "link group", for each pair of fragments. The link-group for a pair of fragments is simply the set of links that could connect the two fragments, where each link connects an object in one fragment's semantic graph to an object in the other fragment's semantic graph. The links are computed using the same set of realization rules that drive the parser and semantic interpreter. They depend on the semantic types of the two objects and on the preposition tag (if any) of the second object. This tag can be relaxed or assumed with a penalty, as we shall see below. For the set of fragments in our example the link-group list is:

```
1a.  FLIGHTS1--- DEST-OF -> BOSTON:TO
1b.  FLIGHTS1--- ORIG-OF -> BOSTON:TO

2a.  FLIGHTS1--- DEST-OF -> DENVER
2b.  FLIGHTS1--- ORIG-OF -> DENVER

3a.  DENVER--- NEARBY-TO -> BOSTON:TO
```

where the links are grouped together in a ordered list according to the fragment-pairs they connect. Since there are three fragments in this example, there are three pairs and thus three groups.

Each link has a set of features which are computed along with the link. The most important is the relational probability of the link, or:

```
P(R,C1,C2)
```

where R is the semantic relation of the link and C1 and C2 are semantic classes of the two argument positions, and C2 may be tagged by a preposition. This is the probability that a pair of objects of type C1 and C2 are linked by by the relation R in an interpretation (instead of linked by some different relation or not linked at all).

A corpus of interpretations generated by hand could be used to determine these probabilities, but we instead use a corpus of 3000 semantic graph interpretations of sentences that our regular parser is able to analyze correctly.

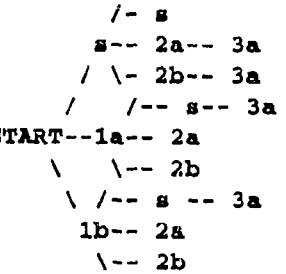
From this corpus, we can determine that the link 1a has a high (.89) probability of connecting a FLIGHT and CITY:TO object, whereas the link 3a has a near zero probability, since the relation NEARBY-CITY-OF occurs very infrequently between two cities.

Links can have other features depending on assumptions made in computing them. For example, a link can be computed by ignoring the prepositional tag of the second object, in which case the link is given the feature "IGNORES-PREP". An example would be 1b above, which ignores the preposition " ". A link can also be computed by assuming a prepositional tag that is not present, giving the link the feature "ASSUMES-PREP", as in 3a, where the preposition "near" is assumed. As we shall see in the next section, these features are also assigned negative weights as penalties, balancing out any higher relational probability the link may have gained from the assumptions made by it.

4.4 Searching the Space of Combinations

In order to structure the search so as to avoid redundant links and duplication of search states, we order the link-group list arbitrarily. Levels of the search tree correspond to elements of the link-group list. At each level, search states are expanded by generating a new state for each link L in the corresponding link-group. L is added to the links already chosen by the parent state to make the links chosen by the new state. An additional "skip" state is generated, which represents the choice not to add any of the links in that link-group. Its links are just those of its parent state.

Below is a portion of the search tree for the problem of connecting our example semantic graph. Each search state is labeled with the link it chose, or with "s" indicating the skip state where no link was chosen:



If a state's chosen links connect all the fragments, the state is said to be "complete", and

no more expansion is done on it. Complete states in the tree above include <a,s,3a>, <1a,2a>, <1b,2a> etc. If the state runs out of link-groups to try, the state is said to be "dead", and no more expansion is done on it. A dead state above is <s,s>.

An important constraint on states is that they not include contradictory links. If a link of the form (R A B'), where R is a single-valued relation, is added to a state that already includes a link of the form (R A B), the clash between B and B' must resolved by attempting to unify the two sub-graphs rooted at these nodes. If unification fails, the state is classified as incoherent, and not expanded further. Examples of incoherent states in the space above are <1a,2a> and <1b,2b>. A "success" state is a complete, coherent state. Some examples are <s2a,3a>, <1a,2b> and <1b,2a>.

Each state is associated with a score, or "cost", which is the sum of the log-probabilities of its chosen links, plus penalties for any other features the state may have, plus an estimated cost for each link, if any, that still has to be added to connect the graph. This score is used to guide an A*-style best-first search through the space.

Link 3a has very low probability, while link 1b has the IGNORES-PREP feature. States choosing these links will therefore have low scores. The other states are incoherent, and the search will therefore produce state <1a,2b> as the best state, with a complete, coherent semantic graph interpretation.

For reasons of space, we have used a very simple example here. Below are actual sentences of the formal test evaluation which the Linker handled correctly:

How much is a coach flight the cheapest coach flight on Southest Airlines
Phoenix till to Milwaukee on Sunday
Los Angeles to Pittsburgh afternoon Tuesday
List flights from Orlando to Tacoma on Saturday of fare basis code of Q
What airline is A S as in Sam
Instead of Saint Louis how about a plane that stops in Denver

4.4.1 Handling Corrections and Ellipsis

A common problem faced by ordinary parsers is speaker disfluency:

Tell me the flights to Denver uhh to Boston

This will produce the fragments "Tell me the flights to Denver" and "to Boston". Since a flight can have only one DEST-OF the fragment "to Boston" cannot be linked according to its most salient DEST-OF interpretation. The alternative would be to ignore the "to" preposition and attempt to link "Boston" as an ORIG-OF with the IGNORE-PREP feature.

This clearly would not produce the correct interpretation, however. The Linker provides an alternative when the clashing value is to the right of the existing value in the string. In this case, graph unification operates in an asymmetrical overwrite mode, in which values to the right in the string replace values to the left. The resulting state receives the combinational feature REPLACEMENT, which is not penalized strongly. If the relational probability of the DEST-OF link is good, it will defeat its IGNORE-PREP rival, as it should. In this way, we are able to integrate some aspects of the disfluency-handling in [19] with the other types of ill-formedness discussed here.

Graph unification with overwriting is also used for ellipsis-handling. In ellipsis-handling mode, the Linker tries to combine the semantic graph of the current utterance into the semantic graph of the previous one. For example, consider the dialogue:

```
''What flights fly to Denver on Wednesday at 3 pm?''

  /----- FLIGHT-OF ---------> FLIGHT1-- QUANTIFIER --> WHAT
FLY1---- DEST-CITY-OF -> DENVER
  \----TIME-OF ---> T1--- D-O-W -> WEDNESDAY
                     \---- T-O-D -- T3 --> HOURS -> 3
                                       \---> AM-PM -> PM
''early Wednesday morning''

   /----- D-O-W --> WEDNESDAY
  T4                                      /-- HOURS -> 8
   \----- T-O-D --> T6 -- BEFORE --> T7
                                       \-- AM-PM -> AM
```

The most plausible link between the two graphs is a TIME-OF between FLY1 and T4, but FLY1 already has a TIME-OF link to T1, and so T1 and T4 must be unified. There is no clash between the D-O-W links, but the graph unification routine, which is extended to do reasoning about certain relations such as "BEFORE" above, detects the clash between the T-O-D links, and and so the T-O-D link of the first graph is replaced with that of the second.

Note that a conventional approach to ellipsis that is based on matching syntactic structures, such as [75], would have difficulty here, since there is little syntactic parallelism in this example. The advantage of our approach is its ability to detect and replace just those components of structure which clash on a semantic level.

4.4.2 Hallucination

Suppose that we have a more telegraphic utterance that does not include the word "flights":

Boston to Denver on Monday Delta

This utterance generates the fragments "Boston", "to Denver", "on Monday" and "Delta". Clearly, no complete set of links can be generated which would fully connect this set, without introducing an object of some other semantic class such as FLIGHT to act as a "hub" between them.

To handle these situations, the Semantic Linker is able to "hallucinate" objects of certain semantic classes, and add link-groups between that hallucinated object and the fragments which are explicitly present. The list of such semantic classes is a domain-dependent parameter of the Linker, and in the ATIS domain comprises just the classes FLIGHT and GROUND-TRANSPORTATION.

A link to a hallucinated object carries a substantial penalty, as it introduces into the discourse an object for which there may be only indirect evidence. The search ordinarily bypasses hallucinated objects, unless alternatives are worse or unavailable.

Note that a reconstructive parsing approach, such as typified by [23] or [72], would potentially have difficulty with this example, as there is not even a fragment which could plausibly act as the syntactic head.

4.5 After Combination - Generating the Logical Form

After the combination phase is complete, we have zero or more success states from which to generate the utterance interpretation. If there is more than one success state, the Linker simply picks the the subset of them with the highest score. In the case that no success states are found, an interpretation may still be generated by "scavenging" through the state space for the best partial connection states found in the course of search.

Once a complete semantic graph has been produced, the Linker must still decide which of the nominal semantics objects are to be displayed to satisfy the user's request – what we term the "topic" of the utterance. Various heuristics are used, including whether the quantifier of the nominal in WH, whether it occurs as an argument to a display verb like "show", and whether it signifies a non-trivial constraint on other nominals.

4.6 Results and Discussion

Our complete system including the Semantic Linker was evaluated in the December 1993 ARPA evaluation. Error rate in this evaluation was defined as F+NA, where F was the percentage of queries answered incorrectly, and NA the percentage of queries not answered at all. Preliminary results indicate that our system received an error rate of 17.8% on the NL test, which was one of the lowest error rates achieved by any of the participating systems.

Our experiments show that using the Semantic Linker reduced our system's error rate on the NL test by 43% (from 31.1% to 17.8%). The no-answer rate NA was dramatically reduced from 18.7% to 2.3%. Just over 80% of this increment of sentences answered were answered correctly, so the Linker showed considerable accuracy.

In conclusion, we have presented a new fallback understanding system that works with semantic representations directly instead of with syntactic structure or task templates. We have also presented a new way to do ellipsis resolution with this component. Furthermore, this system has been proven in formal tests to dramatically improve overall performance.

Several areas of future work are seen. One is the use of automatic training methods to determine feature weights. A second area of future work is the use of relational probabilities and search in the generation of fragments themselves. A third and last area of future work is to more fully integrate the Semantic Linker into the regular parsing mechanism itself, and to investigate ways in which parsing can be viewed as similar to the linking process.

Chapter 5

Written Language Training for Spoken Language Modeling

5.1 Introduction

We attempted to improve recognition accuracy by reducing the inadequacies of the lexicon and language model. Specifically we address the following three problems:

(1) the best size for the lexicon,

(2) conditioning written text for spoken language recognition, and

(3) using additional training outside the text distribution.

We found that increasing the lexicon 20,000 words to 40,000 words reduced the percentage of words outside the vocabulary from over 2% to just 0.2%, thereby decreasing the error rate substantially. The error rate on words already in the vocabulary did not increase substantially. We modified the language model training text by applying rules to simulate the differences between the training text and what people actually said. Finally, we found that using another three years' of training text - even without the appropriate preprocessing, substantially improved the language model. We also tested these approaches on spontaneous news dictation and found similar improvements.

Speech recognition accuracy is affected as much by the language model as by the acoustic model. In general, the word error rate is roughly proportional to the square root of the perplexity of the language model. In addition, in a natural unlimited vocabulary task, a

substantial portion of the word errors come from words that are not even in the recognition vocabulary. These out-of-vocabulary (OOV) words have no chance of being recognized correctly. Thus, our goal is to estimate a good language model from the available training text, and to determine a vocabulary that is likely to cover the test vocabulary.

The straightforward solution to improving the language model might be to increase the complexity of the model (e.g., use a higher order Markov chain) and/or obtain more language model training text. But this by itself will not necessarily provide a better model, especially if the text is not an ideal model of what people will actually say. The simple solution to increase the coverage of the vocabulary is to increase the vocabulary size. But this also increases the word error rate and the computation and size of the recognition process.

In this chapter we consider several simple techniques for improving the power of the language model. First, in Section 3, we explore the effect of increasing the vocabulary size on recognition accuracy in an unlimited vocabulary task. Second, in Section 4, we consider ways to model the differences between the language model training text and the way people actually speak. And third, in Section 5, we show that simply increasing the amount of language model training helps significantly.

5.2 The WSJ Corpus

The November 1993 ARPA Continuous Speech Recognition (CSR) evaluations was based on speech and language taken from the Wall Street Journal (WSJ). The standard language model training text was estimated from about 35 million words of text extracted from the WSJ from 1987 to 1989. The text was normalized (preprocessed) with a model for what words people use to read open text. For example, "$234.56" was *always* assumed to be read as "two hundred thirty four dollars and fifty six cents". "March 13" was always normalized as "March thirteenth" – not "March the thirteenth", nor "March thirteen". And so on.

The original processed text contains about 160,000 unique words. However, many of these are due to misspellings. Therefore, the test corpus was limited to those sentences that consisted only of the most likely 64,000 words. While this vocabulary is still quite large, it has two beneficial effects. First, it greatly reduces the number of misspellings in the texts. Second, it allows implementations to use 2-byte data fields to represent the words rather than having to use 4 bytes.

The "standard" recognition vocabulary was defined as the most likely 20,000 words in the

corpus. Then, the standard language model was defined as a trigram language model estimated specifically for these 20K words. This standard model, provided by Lincoln Laboratory, was to be used for the controlled portion of the recognition tests. In addition, participants were encouraged to generate an improved language model by any means (other than examining the test data).

5.3 Recognition Lexicon

We find that, typically, over 2% of the word occurrences in a development set are not included in the standard 20K-word vocabulary. Naturally, words that are not in the vocabulary cannot be recognized accurately. (At best, we might try to detect that there is one or more unknown words at this point in a sentence, and then attempt to recognize the phoneme sequence, and then guess a possible letter sequence for this phoneme sequence. Unfortunately, in English, even if we could recognize the phonemes perfectly, there are many valid ways to spell a particular phoneme sequence.) However, in addition to this word not being recognize[illegible] often see that one or two words adjacent to this missing word are also mis[illegible] is because the recognition, in choosing a word in its vocabulary, also [illegible]ng context for the following or preceding words. In general, we find that [illegible]rror rate increases by about 1.5 to 2 times the number of out-of-vocabu[illegible]OO[illegible]s.

One simple way to decrea[illegible]e percentage of OOV words is to increase the vocabulary size. But which words should be added? The obvious solution is to add words in order of their relative frequency within the full text corpus. There are several problems that might result from this:

1. The vocabulary might have to be extremely large before the OOV rate is reduced significantly.
2. If the word error rate for the vast majority of the words that are already in the smaller vocabulary increased by even a small amount, it might offset any gain obtained from reducing the OOV rate.
3. The language model probabilities for these additional words would be quite low, which might prevent them from being recognized anyway.

We did not have phonetic pronunciations for all of the 64K words. We sent a list of the (approximately 34K) words for which we had no pronunciations to Boston University.

They found pronunciations for about half (18K) of the words in their (expanded Moby) dictionary. When we added these words to our WSJ dictionary, we had a total of 50K words that we could use for recognition.

The following table shows the percentage of OOV words as a function of the vocabulary size. The measurement was done on the WSJ1 Hub1 "20K" development test which has 2,464 unique words with the total count of 8,227 words. Due to the unavailability of phonetic pronunciations (mentioned above), the final vocabulary size would be the second column.

Top N	Vocab.	#OOV	%
20k	19998	187	2.27
30k	28247	85	1.03
40k	39998	39	0.47
48k	40213	14	0.17
50k	41363	12	0.15
64k	48386	1	0.01

Table 5.1: Out of vocabulary words as a function of vocabulary size

We were somewhat surprised to see that the percentage of OOV words was reduced to only 0.17% when the lexicon included the most likely 40K words – especially given that many of the most likely words were not available because we did not have phonetic pronunciations for them. Thus, it was not necessary to increase the vocabulary above 40K words.

The second worry was that increasing the vocabulary by too much might increase the word error rate due to the increased number of choices. For example, normally, if we double the vocabulary, we might expect an increase in word error rate of about 40%! So we performed an experiment in which we used the standard 20K language model for the 5K development data. We found, to our surprise, that the error rate increased only slightly, from 8.7% to 9.3%. Therefore, we felt confident that we could increase the vocabulary as needed.

We considered possible explanations for the small increase in error due to a larger vocabulary. We realized that the answer was in the language model. In the first case, when we just increase the vocabulary, the new words also have the same probability in the language model as the old words. However, in this case, all the new words that were added had lower probabilities (at least for the unigram model) than the existing words. Let us consider two possibilities that we would not falsely substitute a new word for an old one. If the new word were acoustically similar to one of the words in the test (and therefore

similar to a word in the original vocabulary, then the word would be correctly recognized because the original word would always have a higher language model probability. If, on the other hand, the new word were acoustically very different from the word being spoken, then we might expect that our acoustic models would prevent the new word from being chosen over the old word. While the argument makes some sense, we did not expect the loss for increasing the vocabulary from 5K words to 20K words to be so small.

Finally, the third question is whether the new words would be recognized when they did occur, since (as mentioned above) their language model probabilities were generally low. In fact, we found that, even though the error rate for these new words was higher than for the more likely words, we were still able to recognize about 50% to 70% of them correctly, presumably based largely on the acoustic model. Thus, the net effect of this was to reduce the word error rate by about 1% to 1.5%, absolute.

5.4 Modeling Spoken Language

Another effect that we worked on was the difference between the processed text, as defined by the preprocessor, and the words that people actually used when reading WSJ text. In the pilot WSJ corpus, the subjects were prompted with texts that had already been "normalized", so that there was no ambiguity about how to read a sentence. However, in the WSJ1 corpus, subjects were instructed to read the original texts and to say whatever seemed most appropriate to them. Since the WSJ1 prompting texts were not normalized to deterministic word sequences, subjects showed considerable variability in their reading of the prompting text.

However, the standard language model was derived from the normalized text produced by the preprocessor. This resulted in a mismatch between the language model and the actual word sequences that were spoken. While the preprocessor was quite good at predicting what people said most of the time, there were several cases where people used different words than predicted. For example, the preprocessor predicted that strings like "$234" would be read as "two hundred thirty four dollars". But in fact, most people read this as "two hundred AND thirty four dollars". For another extreme example, the preprocessor's prediction of "10.4" was "ten point four", but the subject (in the WSJ1 development data) read this as "ten and four tenths". There were many other similar examples.

The standard model for the tests was the "nonverbalized punctuation" (NVP) model, which assumes that the readers never speak any of the punctuation words. The other model that had been defined was the "verbalized punctuation" (VP) model, which assumed that *all* of the punctuation was read out loud. This year, the subjects were instructed that they were

free to read the punctuation out loud or not, in whatever way they feel most comfortable. It turns out that people didn't verbalize most punctuation. However, they regularly verbalized quotation marks in many different ways that were all different than the ways predicted by the standard preprocessor.

There were also several words that were read differently by subjects. For example, subjects pronounced abbreviations like, "CORP." and "INC.". While the preprocessor assumed that all abbreviations would be read as full words

We used two methods to model the ways people actually read text. The simpler approach was to include the text of the acoustic training data in the language model training. That is, we simply added the 37K sentence transcriptions from the acoustic training to the 2M sentences of training text. The advantage of this method is that it modeled what people actually said. The system was definitely more likely to recognize words or sequences that were previously impossible The problem with this method was that the amount of transcribed speech was quite small (about 50 times smaller) compared to the original training text. We tried repeating the transcriptions several times, but we found that the effect was not as strong as we would like.

A more powerful approach was to simulate the effects of the different word choices by simple rules which were applied to all of the 35M words of language training text. We chose to use the following rules:

Preprocessed Text	Simulated Text
HUNDRED **[number]**	HUNDRED AND **[number]**
ONE HUNDRED	A HUNDRED
ONE DOLLAR	A DOLLAR
ZERO POINT **[number]**	POINT **[number]**
AND ONE HALF	AND A HALF
AND ONE QUARTER	AND A QUARTER

Thus, for example, if the sentence consists of the pattern "hundred twenty", we repeated the same sentence with "hundred AND twenty".

The result was that about one fifth of the sentences in the original corpus had some change reflecting a difference in the way subjects read the original text Thus, this was equivalent in weight to an equal amount of training text to the original text.

We found that this preprocessing of the text was sufficient to cover most of those cases where the readers said things differently than the predictions. The recognition results showed that the system now usually recognized the new word sequences and abbreviations correctly.

5.5 Increasing the Language Model Training

While 35M words may seem like a lot of data, it is not enough to cover all of the trigrams that are likely to occur in the testing data. So we considered other sources for additional language modeling text. The only easily accessible data available was an additional 3 years (from 1990-1992) of WSJ data from the TIPSTER corpus produced by the Linguistic Data Consortium (LDC).

However, there were two problems with using this data. First, since the test data was known to come from 1987-1989, we were concerned that this might actually hurt performance due to some differences in the topics during that 3-year period. Second, this text had not been normalized with the preprocessor and we did not have available to us the preprocessor that was used to transform the raw text into word sequences.

We decided to use the new text with minimal processing. The text was filtered to remove all tables, captions, numbers, etc. We replaced each initial example of double-quote (") with "QUOTE and the matching token with "UNQUOTE or "ENDQUOTE, which were the most common ways these words were said. No other changes were made. We just used the raw text as it was. One benefit of this was that abbreviations were left as they appeared in the text rather than expanded. Any numbers, dates, dollar amounts, etc, were just considered "unknown" words, and did not contribute to the training. We assumed that we had sufficient examples of numbers in the original text.

We found that adding this additional language training data reduced the error by about 7% of the error, indicating that the original 35 million words was not sufficient for the models we were using. Thus, the addition of plain text, even though it was from a different three years, and had many gaps due to apparent unknown words, still improved the recognition accuracy considerably.

5.6 Results

The following table shows the benefit of the enlarged 40K lexicon and the enhanced language model training on the OOV rate and the word error for the development test and the evaluation test.

Surprisingly, the addition of three year's LM training (from a period post-dating the test data) improved performance on the utterances that were completely inside the vocabulary. Evidently, even the common trigrams are poorly trained with only the 35 million word

	% OOV		% Word Error	
Test Set	20K	40K	20K	40K
Development	2.27	0.17	16.4	12.9
Evaluation	1.83	0.23	14.2	12.2

Table 5.2: Enlarging the lexicon improves OOV rate and error rate

WSJ0 corpus. Overall, our modifications to the lexicon and grammar training reduced the word error by 14–22%.

5.7 Spontaneous Dictation

Another area we investigated was spontaneous dictation. The subjects were primarily former or practicing journalists with some experience at dictation. They were instructed to dictate general and financial news stories that would be appropriate for a newspaper like WSJ. In general, the journalists chose topics of recent interest. This meant that the original language model was often out of date for the subject. As a result, the percentage of OOV words increased (to about 4%), and the language model taken from WSJ text was less appropriate.

The OOV words in the spontaneous data were more likely to be proper nouns from recent events that were not covered by the LM training material. To counter this, we added all (1,028) of the new words that were found in the spontaneous portion of the acoustic training data in WSJ1. This mostly included topical names (e.g., Hillary Rodham, NAFTA, etc.).

In order to account for some of the differences between the read text and the spontaneous text, and to have language model probabilities for the new words, we added the training transcriptions of the spontaneous dictation (about 8K sentences) to the LM training as well.

New weights for the new language model, HMM, and Segmental Neural Network were all optimized on spontaneous development test data. The table below shows that the OOV remains near 1% even after the enlargement to a 41K lexicon.

As can be seen, increasing the vocabulary size from 20K to 40K significantly reduced the OOV rate. It is important to point out that in this case, we did not have the benefit of a word frequency list for spontaneous speech, and that the source of speech had an unlimited

	% OOV			% Word Error	
Test Set	20K	40K	41K	20K	41K
Development	2.9	1.4	0.8	-	21.7
Evaluation	4.8	1.9	1.5	24.7	19.1

Table 5.3: OOV rate and error rate for 41K lexicon

vocabulary. So the reduction in OOV rate is certainly a fair – if not pessimistic – estimate of the real benefit from increasing the vocabulary. Adding the few new words observed in the spontaneous speech also helped somewhat, but not nearly as much. The sample of only 8,000 sentences is clearly not sufficient to find all the new words that people might use. Presumably, if the sample of spontaneous speech were large enough to derive word frequencies, then we could choose a much better list of 40K words with a lower OOV rate.

Overall, the 41K trigram reduces the word error by 23% over the 20K standard trigram on the November '93 CSR S9 evaluation test. We estimate that more than half of this gain was due to the decreased percentage of OOV words, and the remainder was due to the increased language model training, including specific examples of spontaneous dictation.

5.8 Conclusions

We found the following interesting results:

- Expanding the vocabulary with less frequent words does not substantially increase the word error on those words already in the vocabulary, but does eliminate many errors due to OOV words.
- Doubling the amount of language model training text improves the language model, even though the text comes from different years than the test, and even though the text was not preprocessed into proper lexical forms.
- It is possible to improve the quality of the language modeling text by modeling the differences between the predicted reading style and some examples of actual transcriptions.
- Increasing the vocabulary size and language training had a bigger effect on spontaneous speech than it did for read speech.

Chapter 6

HUM - Hidden Understanding Model

6.1 Introduction

uIn this chapter, we describe and evaluate hidden understanding models, a statistical learning approach to natural language understanding. Given a string of words, hidden understanding models determine the most likely meaning for the string. We discuss (1) the problem of representing meaning in this framework, (2) the structure of the statistical model, (3) the process of training the model, and (4) the process of understanding using the model. Finally, we give experimental results, including results on an ARPA evaluation.

Hidden understanding models are an innovative class of statistical mechanisms that, given a string of words, determines the most likely meaning for the string. The overall approach represents a substantial departure from traditional techniques by replacing hand-crafted grammars and rules with statistical models that are automatically learned from examples. Hidden understanding models were primarily motivated by techniques that have been extremely successful in speech recognition, especially hidden Markov models [18]. Related techniques have previously been applied to the problem of identifying concept sequences within a sentence [52]. In addition, the approach contains elements of other natural language processing techniques including semantic grammars [76, 32], augmented transition networks (ATNs) [78], probabilistic parsing [29, 27, 59], and automatic grammar induction [51].

Hidden understanding models are capable of learning a variety of meaning representations, ranging from simple domain-specific representations, to ones at a level of detail and sophistication comparable to current natural language systems. In fact, a hidden

understanding model can be used to produce a representation with essentially the same information content as the semantic graph used by the Delphi system [22], a general purpose NLP system, which utilizes a modified Definite Clause Grammar formalism. This fact made it possible to interface a hidden understanding system to the discourse processing and data- base retrieval components of Delphi to produce a complete end to end system. This hybrid system participated in the 1993 ATIS natural language evaluation. Although only four months old, the scores achieved by the combined system were quite respectable.

Because of differences between language understanding and speech recognition, significant changes are required in the hidden Markov model methodology. Unlike speech, where each phoneme results in a local sequence of spectra, the relation between the meaning of a sentence and the sequence of words is not a simple linear sequential model. Language is inherently nested, with subgroups of concepts within other concepts.

A statistical system for understanding language must take this and other differences into account in its overall design. In principle, we have the following requirements for a hidden understanding system:

- A notational system for expressing meanings.
- A statistical model that is capable of representing meanings and the association between meanings and words.
- An automatic training program which, given pairs of meanings and word sequences, can estimate the parameters of a statistical model.
- An understanding program that can search the statistical model to find the most likely meaning given a word sequence.

Below, we describe solutions for each of these requirements, and describe the relationship of these solutions to other work in stochastic grammars and probabilistic parsing. Finally, we will report on initial experiments with hidden understanding models.

6.2 Expressing Meanings

One of the key requirements for a hidden understanding model is that the meaning representation must be both precise and appropriate for automatic learning techniques.

Specifically, we require a meaning representation that is:

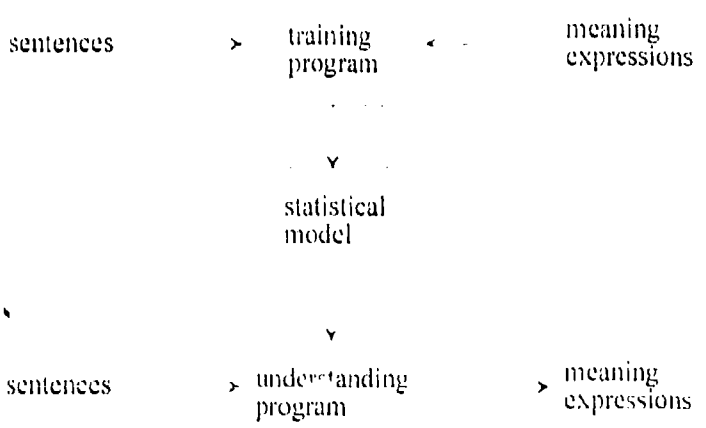


Figure 6.1: The Main Components of a Hidden Understanding System

- Expressive. It must be able to express meanings over the entire range of utterances that are likely to occur in an application.
- Annotatable. It must be possible to produce accurate annotations for a sufficiently large corpus with an acceptable level of human effort.
- Trainable. It must be possible to estimate the model parameters from a reasonable number of training examples.
- Tractable. There must be a computationally tractable algorithm capable of searching the meaning space.

In order to facilitate annotation of a training corpus, meaning expressions should be as simple as possible. Frame based representations, such as the example shown in Figure 6.2, have the advantage that they are relatively simple to understand. A difficulty with this style of representation is that the frames do not align directly to the words of the sentences. In particular, a meaning frame contains few explicit clues as to how the words of a sentence imply the structural characteristics of the frame. Tree structured meaning representations, discussed in the next section, have the advantage that they can be fully aligned to the words of a sentence. The cost is that these tree structured representations are more detailed than their frame based counterparts, thereby requiring greater annotation effort.

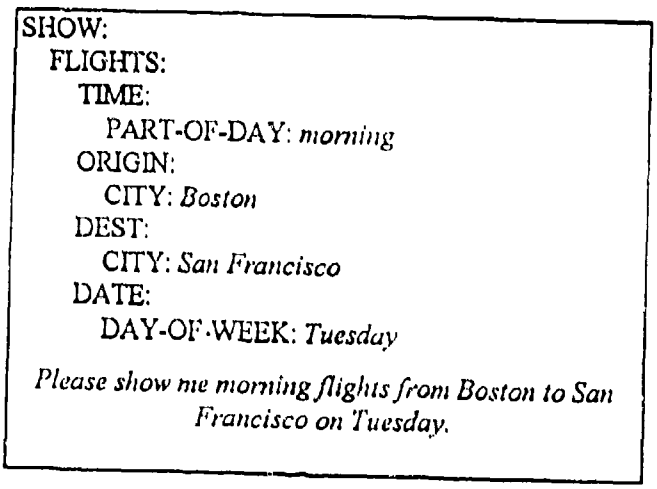


Figure 6.2: A Frame Based Meaning Representation

Fortunately, the techniques developed for tree structured representations can be extended to simpler frame representations as well.

6.2.1 Tree Structured Meaning Representations

The central characteristic of a tree structured representation is that individual concepts appear as nodes in a tree, with component concepts appearing as nodes attached directly below them. For example, the concept of a *flight* in the ATIS domain has component concepts including *airline*, *flight number*, *origin*, and *destination*. These could then form part of the representation for the phrase: *United flight 203 from Dallas to Atlanta*. The use of a hierarchical representation is one characteristic that distinguishes hidden understanding models from earlier work in which meaning is represented by a linear sequence of concepts [52].

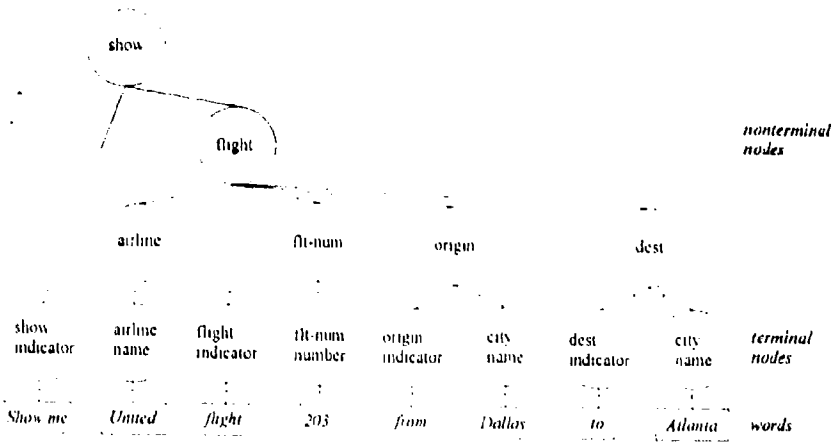


Figure 6.3: A Tree Structured Meaning Representation

A requirement for tree structured representations is that the order of the component concepts must match the order of the words they correspond to. Thus, the representation of the phrase flight 203 to Atlanta from Dallas on United includes the same nodes as the earlier example, but in a different order. For both examples, however, the interpretation is identical.

At the leaves of a meaning tree are the words of the sentence. We distinguish between nodes that appear above other nodes, and those that appear directly above the words. These will be referred to as nonterminal nodes and terminal nodes respectively, forming two disjoint sets. No node has both words and other nodes appearing directly below it.

Figure 6.3 shows an example of a typical meaning tree. In this example, the *flight* node represents the abstract concept of a flight, which is a structured entity that may contain an origin, a destination, and other component concepts. Appearing directly above the word "flight" is a terminal node, which we call a *flight indicator*. This name is chosen to distinguish it from the *flight* node, and also because the word *flight*, in some sense, indicates the presence of a flight concept. Similarly, there are *airline indicators*, *origin indicators*, and *destination indicators*.

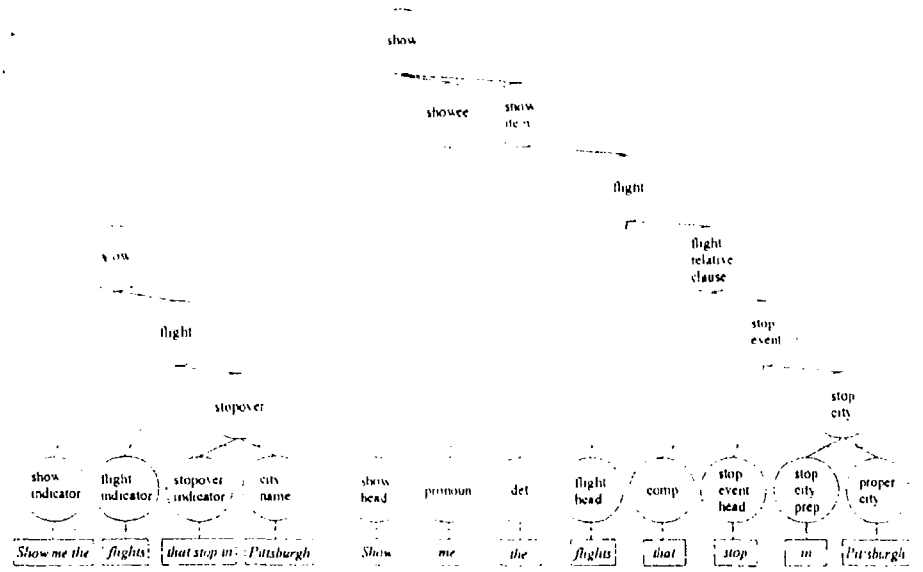


Figure 6.4: A Specialized Sublanguage Analysis and a Full Syntactic Analysis

One view of these tree structured representations is that they are parse trees produced according to a semantic grammar. In this view, the dominance relations of the grammar are predetermined by the annotation schema, while the precedence relations are learned from the training examples.

6.2.2 Alternative Tree Representations

Tree structured meaning expressions can range in complexity from simple special purpose sublanguage representations to the structural equivalent of detailed syntactic parse trees. The possibilities are limited only by two fundamental requirements: (1) semantic concepts must be hierarchically nested within a tree structure, and (2) the sets of terminal and

nonterminal nodes must remain disjoint. Both of these requirements can be satisfied by trees possessing most of the structural characteristics of conventional syntactic parse trees. Since our objective is to model meaning, the nodes must still be labeled to reflect semantic categories. However, additional and augmented labels may be introduced to reflect syntactic categories as well.

Representations of this form contain significantly more internal structure than specialized sublanguage models. This can be seen in the example in Figure 6.4. The specialized sublanguage representation requires only seven nodes, while a full syntactically motivated analysis requires fifteen. The additional nodes are used to distinguish what is being shown to whom, to reflect the fact that the stopover phrase is part of a relative clause, and to determine the internal structure of the relative clause.

One interesting characteristic of these more elaborate trees is their similarity to those produced by classical, linguistically motivated, natural language systems. Thus, a hidden understanding model can serve to replace the part-of-speech tagger, parser, and semantic interpreter of a classical system. Instead of writing grammar and semantic interpretation rules by hand, the training program automatically constructs a statistical model from examples of meaning trees.

Regardless of the details of the tree structure and labels, the components comprising a hidden understanding system remain unchanged. The only difference is in how the system is trained.

6.2.3 Frame Based Representations

One way to think of a frame based meaning is as a partially specified tree in which some words are not accounted for. Nevertheless, a frame representation is a complete meaning representation in the sense that it fully specifies the concepts and structure comprising the meaning. In terms of a tree structured representation, the set of nonterminal nodes is fully specified, while some of the terminal nodes may be omitted.

The missing terminal nodes are said to be hidden, in the sense that every word is required to align to some terminal node, but the alignment is not necessarily given by the meaning frame. These hidden nodes must later be aligned as part of the training process. The general idea is to assign a small number of free terminal nodes (typically one or two) beneath every nonterminal node. These are then free to align to any unassigned words, provided that the overall tree structure is not violated. An EM algorithm (Estimate-Maximize) is used to organize the unassigned terminal nodes into classes that correspond to individual words and phrases, and that bind to particular abstract concepts. Figure 6.5

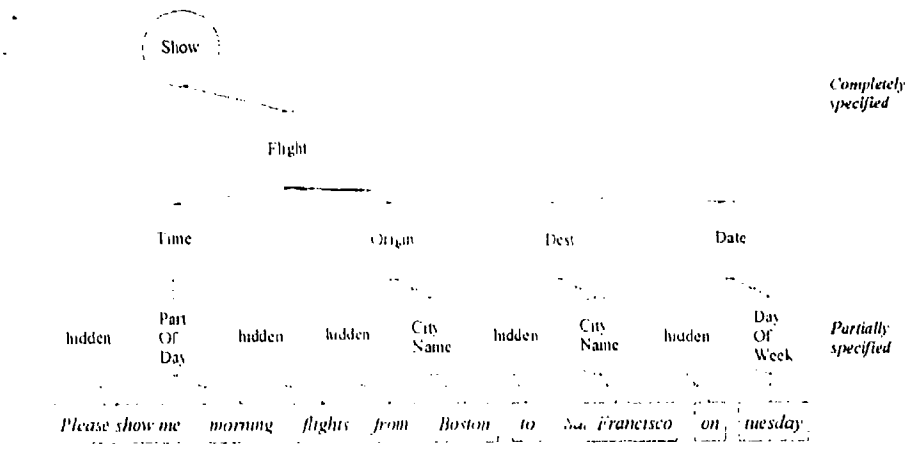


Figure 6.5: A Tree Structure Corresponding to a Frame Representation

shows the complete meaning tree with hidden nodes corresponding to the frame in Figure 6.2.

If we consider tree structured meaning expressions as parse trees which are generated according to some incompletely specified grammar, then the problem of aligning the hidden nodes can be considered as a grammar induction problem. In this way, the problem of aligning the hidden nodes given only a partially specified set of trees is analogous to the problem of fully parsing a training corpus given only a partial bracketing. The difference is that while a partial bracketing determines constituent boundaries that cannot be crossed, a partially specified tree determines structure that must be preserved.

6.3 The Statistical Model

One central characteristic of hidden understanding models is that they are *generative*. From this viewpoint, language is produced by a two component statistical process. The first component chooses the meaning to be expressed, effectively deciding what to say. The second component selects word sequences to express that meaning, effectively deciding how to say it. The first phase is referred to as the *semantic language model*, and can be thought of as a stochastic process that produces meaning expressions selected from a universe of meanings. The second phase is referred to as the *lexical realization model*, and can be thought of as a stochastic process that generates words once a meaning is given.

By analogy with hidden Markov models, we refer to the combination of these two models as a hidden understanding model. The word hidden refers to the fact that only words can be observed. The internal states of each of the two models are unseen and must be inferred from the words. The problem of language understanding, then, is to recover the most likely meaning structure given a sequence of words. More formally, understanding a word sequence W is accomplished by searching among all possible meanings for some meaning M such that $P(M|W)$ is maximized. By Bayes Rule, $P(M|W)$ can be rewritten as:

$$P(M|W) = \frac{P(W|M)P(M)}{P(W)}$$

Now, since $P(W)$ does not depend on M, maximizing $P(M|W)$ is equivalent to maximizing the product $P(W|M)\ P(M)$. However, $P(M|W)$ is simply our lexical realization model, and $P(M)$ is simply our semantic language model. Thus, by searching a combination of these models it is possible to find the maximum likelihood meaning M given word sequence W. Considering the statistical model as a stochastic grammar, the problem of determining M given W is analogous to the problem of finding the most likely derivation for W according to that grammar.

6.3.1 Semantic Language Model

For tree structured meaning representations individual nonterminal nodes determine particular abstract semantic concepts. In the semantic language model, each abstract concept corresponds to a *probabilistic state transition network*. All such networks are then combined into a single *probabilistic recursive transition network*, forming the entire semantic language model.

The network corresponding to a particular abstract concept consists of states for each of its component concepts, together with two extra states that define the entry and exit points. Every component concept is fully connected to every other component concept, with additional paths leading from the entry state to each component concept, and from each component concept to the exit state. Figure 6.6 shows a sample network corresponding to the *flight* concept. Of course, there are many more flight component concepts in the ATIS domain than actually appear in this example. Associated with each arc is a probability value, in a similar fashion to the TINA system [50]. These probabilities have the form $P(State_n|State_{n-1}, Context)$, which is the probability of a taking transition from one state to another within a particular context. Thus, the arc from *origin* to *dest* has probability $P(dest|origin, flight)$, meaning the probability of entering *dest* from *origin* within the context of the *flight* network. Presumably, this probability is relatively high, since people usually mention the destination of a flight directly after mentioning its origin. Conversely, $P(origin|dest, flight)$ is probably low because people don't usually express concepts in that order. Thus, while all paths through the state space are possible, some have much higher probabilities than others.

Within a concept network, component concept states exist for both nonterminal concepts, such as origin, as well as terminal concepts, such as *flight indicator*. Arrows pointing into nonterminal states indicate entries into other networks, while arrows pointing away indicate exits out of those networks. Terminal states correspond to networks as well, although these are determined by the lexical realization model and have a different internal structure. Thus, every meaning tree directly corresponds directly to some particular path through the state space. Figure 6.7 shows a meaning tree and its corresponding path through state space.

Viewed as a grammar, the semantic language model is expressed directly as a collection of networks rather than as a collection of production rules. These networks represent grammatical constraints in a somewhat different fashion than do grammars based on production rules. In this model, constituents may appear beneath nonterminal nodes in any arbitrary order, while preferences for some orderings are determined through the use of probabilities. By contrast, most grammars limit the ordering of constituents to an explicit set which is specified by the grammar rules. The approach taken in the TINA system eliminates many ordering constraints while retaining the local state transition constraints determined by its grammar. We believe that an unconstrained ordering of constraints increases parsing robustness, while the preferences determined by the arc probabilities help minimize overgeneration.

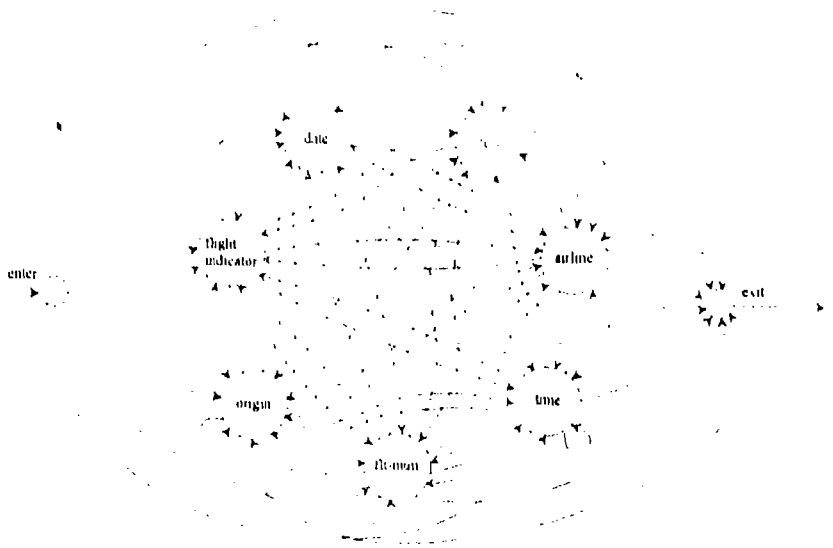


Figure 6.6: A Partial Network Corresponding to the ATIS Flight Concept

6.3.2 Lexical Realization Model

Just as nonterminal tree nodes correspond to networks in the semantic language model, terminal nodes correspond to networks in the lexical realization model. The difference is that semantic language networks specify transition probabilities between states, while lexical realization networks specify transition probabilities between words. Lexical realization probabilities have the form $P(wordn|wordn-1, context)$, which is the probability of taking a transition from one word to another given a particular context. Thus, $P(show|please, show-indicator)$ is the probability that the word *show* follows the word *please* within the context of a *show indicator* phrase. In addition, there are two pseudo-words, *begin* and *end*, which indicate the beginning and ending of phrases. Thus, we have probabilities such as $P(please|*begin*, \text{show-indicator})$, which is the probability that *please* is the first word of a *show indicator* phrase, and $P(*end*|me, show-indicator)$, which is the probability of exiting a show indicator phrase given that the previous word was *me*.

6.4 The Understanding Component

As we have seen, understanding a word string W requires finding a meaning M such that the probability $P(W|M)P(M)$ is maximized. Since, the semantic language model and the lexical realization model are both probabilistic networks, $P(W|M)P(M)$ is the probability of a particular path through the combined network. Thus, the problem of understanding is to find the highest probability path among all possible paths, where the probability of a path is the product of all the transition probabilities along that path.

$$P(Path) = \prod_{t \in Path} u(n) = \begin{bmatrix} P(state_n|state_{n-1}, context) & \text{if } t \text{ in Semantic Language Model} \\ P(word_n|word_{n-1}, context) & \text{if } t \text{ in Lexical Realization Model} \end{bmatrix} \quad (6.1)$$

Thus far, we have discussed the need to search among all meanings for one with a maximal probability. In fact, if it were necessary to search every path through the combined network individually, the algorithm would require exponential time with respect to sentence length. Fortunately, this can be drastically reduced by combining the probability computation of common subpaths through dynamic programming. In particular, because our meaning representation aligns to the words, the search can be efficiently performed using the well-known Viterbi [74] algorithm.

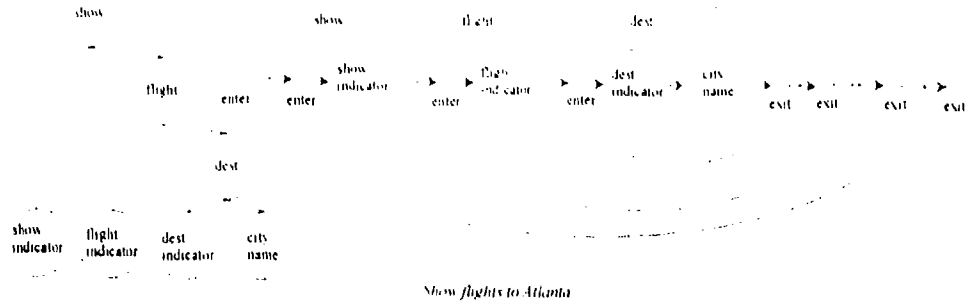


Figure 6.7: A Meaning Tree and its Corresponding Path Through State Space

Since our underlying model is a recursive transition network, the states for the Viterbi search must be allocated dynamically as the search proceeds. In addition, it is necessary to prune very low probability paths in order to keep the computation tractable. We have developed an elegant algorithm that integrates state allocation, Viterbi search, and pruning all within a single traversal of a tree- like data structure. In this algorithm, each of the set of currently active states is represented as a node in a tree. New nodes are added to the tree as the computation pushes into new subnetworks that are not currently active. Stored at each node is the probability of the most likely path reaching that state, together with a backpointer sufficient to recreate the path later if needed. Whenever the probability of all states in a subtree falls below the threshold specified by the beam width, the entire subtree is pruned away.

6.5 The Training Component

In order to train the statistical model, we must estimate transition probabilities for the semantic language model and lexical realization model. In the case of fully specified meaning trees, each meaning tree can be straightforwardly converted into a path through state space. Then, by counting occurrence and transition frequencies along those paths, it is possible to form simple estimates of the transition probabilities. Let $C(statem, contexts)$ denote the number of times statem has occurred in contexts, and let $C(staten|statem, contexts)$ denote the number of times that this condition has led to a transition to state staten. Similarly, define counts $C(wordm, contextl)$ and $C(wordn|wordm, contextl)$. Then, a direct estimate of the probabilities is given by:

$$\hat{P}(state_n|state_m, context) = \frac{C(state_n|state_m, context)}{C(state_m, context)}$$

and

$$\hat{P}(word_n|word_m, context) = \frac{C(word_n|word_m, context)}{C(word_m, context)}$$

In order to obtain robust estimates, these simple estimates are smoothed with *backed-off* estimates [31], using techniques similar to those used in speech recognition [38, 55]. Thus, $P(staten|statem, context)$ is smoothed with $P(staten|context)$, and $P(wordn|wordm, context)$ is smoothed with $P(wordn|context)$. Robustness is further increased through word classes. For example, *Boston* and *San Francisco* are both members of the class of cities.

In the case of frame based representations, it is not always possible to construct an exact path through the state space corresponding to a meaning representation. Nevertheless, since

frames are treated as partially specified trees, most of the path can be reconstructed, with some portions undetermined. Then, the partial path can be used to constrain a gradient descent search, called the forward- backward algorithm [18] for estimating the model parameters. This algorithm is an iterative procedure for adjusting the model parameters so as to increase the likelihood of generating the training data, and is an instance of the well-known class called EM (Estimate-Maximize) algorithms.

6.6 Experimental Results

We have implemented a hidden understanding system and performed a variety of experiments. In addition, we participated in the 1993 ARPA ATIS NL evaluation. One experiment involved a 1000 sentence ATIS corpus, annotated according to a simple specialized sublanguage model. The annotation effort was split between two annotators, one of whom was a system developer, while the other was not. To annotate the training data, we used a bootstrapping process in which only the first 100 sentences were annotated strictly by hand.

Thereafter, we worked in cycles of:

1. Running the training program using all available annotated data.
2. Running the understanding component to annotate new sentences.
3. Hand correcting the new annotations.

Annotating in this way, we found that a single annotator could produce 200 sentences per day. We then extracted the first 100 sentences as a test set, and trained the system on the remaining 900 sentences. The results were as follows:

- 61% matched exactly.
- 21% had correct meanings, but did not match exactly.
- 28% had the wrong meaning.

Another experiment involved a 6000 sentence ATIS corpus, annotated according to a more sophisticated meaning model. In this experiment, the Delphi system automatically produced the annotation by printing out its own internal representation for each sentence, converted into a more readable form. In order to maintain high quality annotations, we

used only sentences for which Delphi produced a complete parse, and for which it also retrieved a correct answer from the database. We then removed 300 sentences as a test set, and trained the system on the remaining 5700. The results were as follows:

- 85% matched exactly,
- 8% had correct meanings, but did not match exactly,
- 7% had the wrong meaning.

For the ARPA evaluation, we coupled our hidden understanding system to the discourse and backend components of the Delphi. Using the entire 6000 sentence corpus described above as training data, the system produced a score of 26% simple error on the ATIS NL evaluation. By examining the errors, we have reached the conclusion that nearly half are due to simple programming issues, especially in the interface between Delphi and the hidden understanding system. In fact, the interface was still incomplete at the time of the evaluation.

We have just begun a series of experiments using frame based annotations, and are continuing to refine our techniques. In a preliminary test involving a small corpus of 588 ATIS sentences, the system correctly aligned the hidden states for over 95% of the sentences in the corpus.

6.7 Limitations

Several limitations to our current approach are worth noting. In a small number of cases, linguistic movement phenomena make it difficult to align the words of a sentence to any tree structured meaning expression without introducing crossings. In most cases, we have been able to work around this problem by introducing minor changes in our annotation such that the tree structure is maintained. A second limitation, due to the local nature of the model, is an inability to handle nonlocal phenomena such as conference. Finally, in some cases the meaning of a sentence depends strongly upon the discourse state, which is beyond the scope of the current model.

6.8 Conclusions

We have demonstrated the possibility of automatically learning semantic representations directly from a training corpus through the application of statistical techniques. Empirical results, including the results of an ARPA evaluation, indicate that these techniques are capable of relatively high levels of performance.

While hidden understanding models are based primarily on the concepts of hidden Markov models, we have also shown their relationship to other work in stochastic grammars and probabilistic parsing.

Finally, we have noted some limitations to our current approach. We view each of these limitations as opportunities for further research and exploration.

Bibliography

[1] Ades, A. E. and Steedman, M. J., "On the Order of Words," *Linguistics and Philosophy* 4.4.3, 1982, pp. 517–558, 1982.

[2] Alshawi, H. (ed.), "The Core Language Engine", MIT Press, Cambridge, 1992.

[3] Austin, S, G. Zavaliagkos, J. Makhoul and R. Schwartz, "Improving State-of-the-Art Continuous Speech Recognition Systems Using the N-Best Paradigm with Neural Networks", in Proceedings of the 5th DARPA Speech and NL Workshop at Arden House, February 23-26, 1992, Morgan Kaufmann, 1992.

[4] Austin, S., J, Makhoul and R, Schwartz, G. Zavaliagkos, "Speech Recognition Using Segmental Neural Nets," *International Conference on Acoust., Speech and Signal Processing*, 1992.

[5] Austin, S., Schwartz, R., and P. Placeway, "The Forward-Backward Search Algorithm," *International Conference on Acoust., Speech and Signal Processing*, Toronto, Canada, pp. 697-700, 1991.

[6] Bates, M., "Beginning to Port a Spoken Language Database Interface," *4th Annual Dual Use Technologies and Applications Conference*, Utica, NY, May 1994.

[7] Bates, M., "Models of Natural Language Understanding," *Voice Communication Between Humans and Machines*, L. Rabiner (Ed.), 1994.

[8] Bates, M, R. Bobrow and R. Weischedel, "Critical Challenges for NLP," *Challenges in Natural Language Processing*, Cambridge University Press, 1993.

[9] Bates, M. et.al., "The BBN/HARC Spoken Language Understanding System," *Proceedings of IEEE ICASSP-93*, Minneapolis, MN, pp. 111-114, vol. II, April 1993.

[10] Bates, M, and R. Weischedel, (eds.), Challenges in Natural Language Processing, Cambridge University Press, 1993.

[11] Bates, M. (ed.) *Proceedings of the Human L uage Workshop*, Merrill Lynch Conference Center, Plainsboro, NJ, Morgan Ka mann Publishers, 1993.

[12] Bates, M., R. Bobrow, P. Fung, R. Ingria, F. Kubala, J. Makhoul, L. Nguyen, R. Schwartz, D. Stallard, "The BBN/HARC Spoken Language Understanding System," *International Conference on Acoust., Speech and Signal Processing*, Minneapolis, MN, April 1993, pp. 111-114, vol. II.

[13] Bates, M., R. Bobrow, P. Fung, R. Ingria, F. Kubala, J. Makhoul, L. Nguyen, R. Schwartz, and D. Stallard, "Design and Perform. nce of Hare, the BBN Spoken Language Understanding System," *Proceedings of the International Conference on Spoken Language Processing*, Alberta, Canada, October 1992.

[14] Bates, M.,et.al, "Commercial, Isolated Word Speech Recognition Integrated with NL Processing for Intelligence Applications," *IEEE 1992 C3I Technology and Applications Conference*, Rome, NY, June 1-4, 1992.

[15] Bates, M. and S. Boisen, "A Developing Methodology for the Evaluation of Spoken Language Systems," *Workshop on Evaluation of Natural Language Processing Systems*, 29th Annual Meeting of the Association for Computational Linguistics, Berkeley, CA, June 19-21, 1991.

[16] Bates, M., "Rapid Porting of the Parlance Natural Language Interface," Proc. DARPA Speech and Natural Language Workshop, Philadelphia, PA, Morgan Kaufmann Publishers, February 1989, pp. 83-88.

[17] Bates, M., D. Stallard, and M. Moser, "The IRUS Transportable Natural Language Database Interface," Expert Database Systems, Cummings Publishing Company, Menlo Park, CA. 1985.

[18] Baum, E., "An Inequality and Associated Maximization Technique in Statistical Estimation of Probabilistic Functions of Markov Processes," Inequalities 3:1-8, 1972.

[19] Bear, J., J. Dowding and E. Shriberg, "Block Integrating Multiple Knowledge Sources for Detection and Correction of Repairs in Human-Computer Dialog," *Proceedings 30th ACL Meeting*, 1992.

[20] Bobrow, R., "Statistical Agenda Parsing," *Proc. of the DARPA Speech and Natural Language Workshop*, Morgan Kaufmann Publishers, Feb. 1991, pp. 222–224.

[21] Bobrow, R., R. Ingria, and D. Stallard, "Syntactic/Semantic Coupling in the BBN DELPHI System," in *Proceedings of the 5th DARPA Speech and NL Workshop at Arden House*, February 23-26, 1992, Morgan Kaufmann, 1992.

[22] Bobrow, R. Ingria, and D. Stallard, "Syntac and Semantic Knowledge in the DELPHI Unific tion Grammar," *Proceedings, ech and Natural Language Worksho*, pp. 230-236, June 1990.

[23] Bobrow R., D. Stallard, "Fragment Processing in the DELPHI System," *Proc. of the DARPA Speech and Natural Language Workshop*, Morgan Kaufmann Publishers, Feb. 1992.

[24] Boisen, S., and M. Bates "A Practical Methodology for the Evaluation of Spoken Language Systems," *Proceedings of the 3rd Co ference on Applied Natural Language Processing*, ACL, Trento, Italy, April, 1992.

[25] Chow, Y., M. Dunham, O Kimball, M. Krasner, G.F. Kubala, J. Makhoul, P. Price, S. Roucos, and R. Schwartz , "BYBLOS: The BBN Continuous Speech Recognition System," *IEEE ICASSP-87*, pp. 89-92

[26] Chow, Y-L. and R.M. Schwartz, "The N-Best Algorithm: An Efficient Procedure for Finding Top N Sentence Hypotheses," *Proc. of the DARPA Speech and Natural Language Workshop*, Morgan Kaufmann Publishers, Oct. 1989, pp. 199–202. Also in ICASSP-90, April 1990, Albuquerque S2.12, pp. 81-84.

[27] Chitrao, and R. Grishman, "Statistical Parsing of Messages," *Proceedings, Speech and Natural Language Workshop*, pp. 263-276, Morgan Kaufmann Publishers, June 1990.

[28] Dowding, J. et al., "Gemini: A Natural Language Understanding System for Spoken Language Understanding," *Proceedings of the 31st Annual Meeting of the Association for Computational Linguistics*, Columbus, OH.

[29] Fujisaki, F. Jelinek, J. Cocke, E. Black, T. Nishino, A "Probabilistic Parsing Method for Sentence Disambiguation,"*International Parsing Workshop*, pp. 85-90, 1989

[30] Gish, H. "A Minimum Classification Error, Maximum Likelihood, Neural Network," *International Conference on Acoust., Speech and Signal Processing*, 1992.

[31] Good, "The Population Frequencies of Species and the Estimation of Population Parameters," *Biometrika 40*, pp. 237-264, 1953.

[32] Hendrix, G.G., "Semantic Aspects of Translation, Understanding Spoken Language," pp. 193-226, New York, Elsevier, 1978.

[33] Hirschman L.et. al., "Multi-Site Data Collection for a Spoken Language Corpus," *Proceedings of the ICSLP-92 Conference, (International Conference on Spoke Language Processing*, Banff, Alberta, Canada, 1992.

[34] R.J. Ingria, and L. Ramshaw, "Porting to New Domains Using the Learner," *Proceedings of the Speech and NL Workshop*, Cape Cod, MA, Morgan Kaufmann Publishers, pages 241-244. Oct. 1989.

[35] Ingria, R., "DARPA Common Lexicon Progress Report," *5th DARPA Speech and NL Workshop*, Arden House, February 23-26, 1992.

[36] Jackson, E., D.Appelt, J. Bear, R. Moore, and A. Podlozny, "A Template Matcher for Robust NL Interpretation," *Proceedings Speech and Natural Language Workshop*, February 1991.

[37] Katz, "Estimation of Probabilities from Sparse Data for the Language Model Component of a Speech Recognizer," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, Vol. ASSP-35, pp. 400-401, 1987.

[38] Katz, S., "Estimation of Probabilities from Sparse Data for the Language Model Component of a Speech Recognizer," *IEEE Trans. on ASSP*, Mar. 1987, Vol. 35, No. 3.

[39] Kimball, O., M. Ostendorf and R. Rohlicek, "Recognition Using Classification and Segmentation Scoring," in *Proceedings of the 5th DARPA Speech and NL Workshop at Arden House*, February 23-26, 1992, Morgan Kaufmann, 1992.

[40] Kubala, F. et al, "BBN BYBLOS and HARC February 1992 ATIS Benchmark Results," *5th DARPA Speech & NL Workshop at Arden House*, February 23-26, 1992.

[41] Kubala, F., C. Barry, M. Bates, R. Bobrow, P. Fung, R. Ingria, J. Makhoul, L. Nguyen, R. Schwartz, D. Stallard, "BBN BYBLOS and HARC February 1992 ATIS Benchmark Results," *Proc. of the DARPA Speech and Natural Language Workshop*, Morgan Kaufmann Publishers, Feb. 1992.

[42] Linebarger, M.C., L.M. Norton, and D.A. Dahl, "A Portable Approach to Last Resort Parsing and Interpretation," *Proceedings Human Language Technology Conference*, Morgan Kaufmann, 1993.

[43] Miller, S., R. Bobrow, R. Ingria, and R. Schwartz, "Hidden Understanding Models of Natural Language," 32nd Annual Meeting of the Assoc. Computational Linguistics, Morristown, NJ, June 1994.

[44] MADCOW, "Multi-Site Data Collection for a Spoken Language Corpus," *Proc. of the DARPA Speech and Natural Language Workshop*, Morgan Kaufmann Publishers, Feb. 1992.

[45] Makhoul, J., "State of the Art in Continuous Speech Recognition," *Voice Communication Between Humans and Machines*, L. Rabiner (Ed.), 1994.

[46] Miller, S., R. Bobrow, R. Ingria, and R. Schwartz, "A Preliminary Investigation of Hidden Understanding Models," Proc. ARPA Human Language Technology Workshop, Plainsboro, NJ, March 1994.

[47] Ng, K., H. Gish and R. Rohlicek, "Robust Mapping of Noisy Speech Parameters for HMM Work Spotting," *International Conference on Acoust., Speech and Signal Processing*, 1992.

[48] Nguyen, L., R. Schwartz, Y. Zhao, and G. Zavaliagkos, "Is N-Best Dead?" Proc. ARPA HLT Workshop, Plainsboro, NJ, Morgan Kaufmann, March 1994

[49] Nguyen, L., R. Schwartz, F. Kubala, and P. Placeway, "Search Algorithms for Software-Only Real-Time Recognition with Very Large Vocabularies," *In Proceedings of the Human Language Workshop*, Merrill Lynch Conference Center, Plainsboro, NJ, Morgan Kaufmann Publishers, 1993.

[50] Pallet, D., J. Fiscus, W. Fisher and J. Garofolo, "Benchmark Tests for the Spoken Language Program," *Proceedings Human Language Technology Conference*, Morgan Kaufmann, 1993.

[51] Pereira and Y. Schabes, "Inside-Outside Reestimation from Partially Bracketed Corpora," *Proceedings of the 30th Annual Meeting of the Association for Computational Linguistics*, pp.128-135, Newark, Delaware, 1992.

[52] Pieraccini, R., E. Levin, and C. Lee, "Stochastic Representation of Conceptual Structure in the ATIS Task: DARPA Speech and Natural Language Workshop," pp. 121-124, Feb. 1991.

[53] Pereira, F.C.N. and D.H.D. Warren, "Definite Clause Grammars for Language Analysis—A Survey of the Formalism and a Comparison with Augmented Transition Networks," *Artificial Intelligence* 13, pp. 231–278, 1980.

[54] Placeway, R. Schwartz, P. Fung, L. Nguyen, "The Estimation of Powerful Language Models from Small and Large Corpora," *International Conference on Acoust., Speech and Signal Processing*, II:33-36.

[55] Placeway, P., Schwartz, R., Fung, P., and L. Nguyen, "The Estimation of Powerful Language Models from Small and Large Corpora," *International Conference on Acoust., Speech and Signal Processing*, Minneapolis, MN, April, 1993.

[56] Rohlicek, R., M. Ostendorf, "A Bayesian Approach to Speaker Adaptation for the Stochastic Segment Model," *International Conference on Acoust., Speech and Signal Processing*, 1992.

[57] Rohlicek, R., D. Ayuso, M. Bates, R. Bobro... V. Boulanger, H. Gish, P. Jeanrenaud, M. Meteer and M. Siu, "Gisting Conversational Speech," *International Conference on Acoust., Speech and Signal Processing*, 1992.

[58] Schabes, Y., A. Abeille, and A.K. Joshi, "Parsing Strategies with 'Lexicalized' Grammars': Application to Tree Adjoining Grammars," *COLING Budapest: PROCEEDINGS of the 12th International Conference on Computational Linguistics*, Association for Computational Linguistics, Morristown, NJ, pp. 578–583, 1988.

[59] Seneff, T., "A Natural Language System for Spoken Language Applications, Computational Linguistics ," Vol. 18, Number 1, pp. 61-86, March 1992.

[60] Schwartz, R. et al, "Continuous Speech Research Final Report," BBN Systems and Technologies, September, 1994.

[61] Schwartz, R., L. Nguyen, F. Kubala, G. Chou, G. Zavaliagkos, and J. Makhoul, "On Using Written Language Training Data for Spoken Language Modeling," Proc. ARPA HLT Workshop, Plainsboro, NJ, March 1994.

[62] Schwartz, R., A. Anastasakos, F. Kubala, J. Makhoul, L. Nguyen, G. Zavaliagakos, "Comparative Experiments on Large Vocabulary Speech Recognition," *Proceedings of the Human Language Workshop*, Merrill Lynch Conference Center, Plainsboro, NJ, Morgan Kaufmann Publishers, 1993.

[63] Schwartz, R., S. Austin, Kubala, F., and J. Makhoul, "New Uses for the N-Best Sentence Hypotheses Within the BYBLOS Speech Recognition System," *International Conference on Acoust., Speech and Signal Processing*, San Francisco, CA, 1992.

[64] Schwartz, R. and S. Austin, "A Comparison Of Several Approximate Algorithms for Finding Multiple (N-Best) Sentence Hypotheses," *International Conference on Acoust., Speech and Signal Processing*, Toronto, Canada, pp. 701-704, 1991.

[65] Schwartz, R.M., and S.A. Austin, "Efficient, High-Performance Algorithms for N-Best Search," *Proceedings of the DARPA Speech and Natural Language Workshop*, Morgan Kaufmann Publishers, Inc., Jun. 1990.

[66] Shieber, S. M., "An Introduction to Unification-Based Approaches to Grammar," *Center for the Study of Language and Information*, Stanford, CA, 1986.

[67] Siu, M.-H., G. Yu and H. Gish, "An Unsupervised Sequential Learning Algorithm for the Segmentation of Speech Waveforms with Multiple Speakers," *International Conference on Acoust., Speech and Signal Processing*, 1992.

[68] Stallard, D.,"Two Kinds of Metonymy," 31st Annual Meeting Assoc. Computational Linguistics, Columbus, Ohio, June 1993.

[69] Stallard, D., "Unification-Based Semantic Interpretation in the BBN Spoken Language System," *Proc. of the DARPA Speech and Natural Language Workshop*, Morgan Kaufmann, Publishers, Oct. 1989, pp. 39–46.

[70] Stallard, D. and R. Bobrow, "The Semantic Linker – a New Fragment Combining Method," *Proceedings Human Language Technology Workshop* Morgan Kaufmann, March 1993.

[71] Stallard, D. and R.Bobrow, "The Mapping Unit Approach to Subcategorization," *Proceedings Speech and Natural Language Workshop*, March 1991.

[72] Seneff, S., "A Relaxation Method for Understanding Spontaneous Speech Utterances," *Proceedings Speech and Natural Language Workshop*, February 1992.

[73] Thompson H., (ed.) "An Evolving Methodology for the Evaluation of Spoken Language Systems," *The Strategic Role of Evaluation in NL Processing and Speech Technology*, a record of a workshop sponsored by the ESPRIT Working Group on Dialogue and Discourse, the European Network of Excellence in Language and Speech, and the Human Communication Research Centre of the University of Edinburgh, Scotland, 1992.

[74] Viterbi, J., "Error Bounds for Convolutional Codes and an Asympotically Optimum Decoding Algorithm," *IEEE Transactions on Information Theory* IT-13(2):260- 269, April 1967

[75] Walker, D.E., (ed.)., "Understanding Spoken Language," New York: Elsevier North-Holland, 1978.

[76] Waltz, D.L., "An English Language Question Answering System for a Large Relational Database," *Communications of the ACM* 21(7):526-39, 1978.

[77] Woods, WA., R.A. Kaplan, and B. Nash-Webber, The Lunar Sciences Natural Language Information System: Final Report, Report 2378, Bolt, Beranek and Newman, Cambridge, MA, 1978.

[78] Woods, W.A., "Transition Network Grammars for Natural Language Analysis," *Communications of the ACM* 13(10):591-606, 1970.